

A Tight Combinatorial Algorithm for Submodular Maximization Subject to a Matroid Constraint

Yuval Filmus and Justin Ward

March 2, 2013

Abstract

We present an optimal, combinatorial $1 - 1/e$ approximation algorithm for monotone submodular optimization over a matroid constraint. Compared to the continuous greedy algorithm (Calinescu, Chekuri, Pál and Vondrák, 2008), our algorithm is extremely simple and requires no rounding. It consists of the greedy algorithm followed by local search. Both phases are run not on the actual objective function, but on a related non-oblivious potential function, which is also monotone submodular. Our algorithm runs in randomized time $\tilde{O}(n^7 u^2)$, where n is the rank of the given matroid and u is the size of its ground set. We additionally obtain a $1 - 1/e - O(\epsilon)$ approximation algorithm running in randomized time $\tilde{O}(\epsilon^{-3} n^4 u)$. For matroids in which $n = o(u)$, this improves on the runtime of the continuous greedy algorithm. The improvement is due primarily to the time required by the rounding phase, which we avoid altogether. Furthermore, the independence of our algorithm from rounding techniques suggests that our general approach may be helpful in contexts such as monotone submodular maximization subject to multiple matroid constraints.

In our previous work on maximum coverage (Filmus and Ward, 2011), the potential function gives more weight to elements covered multiple times. We generalize this approach from coverage functions to arbitrary monotone submodular functions. When the objective function is a coverage function, both definitions of the potential function coincide. The parameters used to define the potential function are closely related to Padé approximants of e^x evaluated at $x = 1$. We use this connection to determine the approximation ratio of the algorithm.

Our approach generalizes to the case where the monotone submodular function has restricted curvature. For any curvature c , we adapt our algorithm to produce a $(1 - e^{-c})/c$ approximation. This result complements results of Vondrák (2008), who has shown that the continuous greedy algorithm produces a $(1 - e^{-c})/c$ approximation when the objective function has curvature c . He has also proved that achieving any better approximation ratio is impossible in the value oracle model.

This paper supersedes the authors' FOCS 2012 paper. A reworking of the proof can be found on the first author's homepage.

1 Introduction

In this paper, we consider the problem of maximizing a monotone submodular function f , subject to a single matroid constraint. Formally, let \mathcal{U} be a set of elements and let $f: 2^{\mathcal{U}} \rightarrow \mathbb{R}_{\geq 0}$ be a function assigning a value to each subset of \mathcal{U} . We say that f is *submodular* if

$$f(A) + f(B) \geq f(A \cup B) + f(A \cap B)$$

for all $A, B \subseteq \mathcal{U}$. If additionally, $f(A) \leq f(B)$ whenever $A \subseteq B$, we say that f is *monotone submodular*. Submodular functions exhibit (and are, in fact, alternately characterized by) the property of diminishing returns—if f is submodular then $f(A \cup \{x\}) - f(A) \leq f(B \cup \{x\}) - f(B)$ for all $B \subseteq A$. Hence, they are useful for modeling various economic and game-theoretic scenarios, as well as various combinatorial problems. In a general monotone submodular maximization problem, we are given a value oracle for f and a membership oracle for some distinguished collection $\mathcal{I} \subseteq 2^{\mathcal{U}}$ of *feasible sets*, and our goal is to find a member of \mathcal{I} that maximizes the value of f . We assume further that f is *normalized* so that $f(\emptyset) = 0$.

We consider the restricted setting in which the collection \mathcal{I} is a matroid. Matroids are intimately connected to combinatorial optimization: the problem of optimizing a linear function over a hereditary set system (a set system closed under taking subsets) is solved optimally for all possible functions by the standard greedy algorithm if and only if the set system is a matroid [27, 8].

In the case of a *monotone submodular* objective function, the standard greedy algorithm, which takes at each step the element yielding the largest increase in f while maintaining independence, is (only) a 2-approximation [15]. Recently, Calinescu et al. [5, 28, 6] have developed a $(1 - 1/e)$ -approximation for this problem via the *continuous greedy algorithm*, which is essentially a steepest descent algorithm running in continuous time (in practice, a suitably discretized version is used), producing a fractional solution. This solution is rounded using pipage rounding [1] or swap rounding [7].

Feige [9] shows that improving the bound $(1 - 1/e)$ is NP-hard. Nemhauser and Wolsey [24] show that any improvement over $(1 - 1/e)$ requires an exponential number of queries in the value oracle setting.

Following Vondrák [29], we also consider the case when f has restricted curvature. We say that f has *restricted curvature* c with respect to \mathcal{I} if for any two non-empty and disjoint independent sets A, B ,

$$f(A \cup B) \geq f(A) + (1 - c)f(B).$$

When $c = 1$, this is a restatement of monotonicity. Vondrák [29] has shown that the continuous greedy algorithm produces a $(1 - e^{-c})/c$ approximation when f has restricted curvature c . Furthermore, he has shown that any improvement over $(1 - e^{-c})/c$ requires an exponential number of queries in the value oracle setting.

Our definition of curvature differs from the usual (non-restricted) definition of curvature in that we only consider sets A and B that are independent, while the general definition requires that the above inequality hold for all (not necessarily independent) sets A and B . Therefore, every function with curvature at most c also has restricted curvature at most c and so our algorithm applies to a *wider* class of functions than those with non-restricted curvature at most c .

1.1 Our contribution

In this paper, we propose a conceptually simple randomized polynomial time local search algorithm for the problem of monotone submodular matroid maximization. Like the continuous greedy algorithm, our algorithm delivers the optimal $(1 - 1/e)$ -approximation. However, unlike the continuous greedy algorithm, our algorithm is entirely combinatorial, in the sense that it deals only with integral solutions to the problem and hence involves no rounding procedure. As such, we believe that the algorithm may serve as a gateway to further improved algorithms in contexts where pipage rounding and swap rounding break down, such as submodular maximization subject to multiple matroid constraints.

Our main results are a combinatorial $1 - 1/e - \epsilon$ approximation algorithm for monotone submodular matroid maximization, running in randomized time $\tilde{O}(\epsilon^{-3}n^4u)$ and a combinatorial $1 - 1/e$ approximation algorithm running in randomized time $\tilde{O}(n^7u^2)$, where n is the rank of the given matroid and u is the size of its ground set. We compare the runtime of our algorithms to the continuous greedy algorithm (our estimate of the continuous greedy algorithm's runtime appears in Appendix B). We show that, while our algorithms' runtimes are a factor of $\tilde{O}(n)$ greater than the continuous greedy algorithm's initial continuous greedy phase, they have greatly improved dependence on u , when compared to the continuous greedy algorithm's rounding phase (either pipage rounding or swap rounding)¹. Thus, our algorithms attain a significant runtime improvement over the continuous greedy algorithm when $n = o(u)$.

Our algorithm further generalizes to the case in which the submodular function has restricted curvature c . In this case the approximation ratios obtained are $(1 - e^{-c})/c - \epsilon$ and $(1 - e^{-c})/c$, respectively, again matching the performance of the continuous greedy algorithm [29]. Unlike the continuous greedy algorithm, our algorithm requires knowledge of c . However, by enumerating over values of c we are able to obtain a combinatorial $(1 - e^{-c})/c$ algorithm even in the case that f 's curvature is not given (assuming it is bounded away from zero). Vondrák's make use of restricted notions of curvature, and so apply to a class of functions slightly larger than those of curvature at most c .

Our algorithmic approach is based on *non-oblivious local search*, a technique first proposed by Alimonti [2] and by Khanna, Motwani, Sudan and Vazirani [20]. In classical (or, *oblivious*) local search, the algorithm starts at an arbitrary solution, and proceeds by iteratively making small changes that improve the objective function, until no such improvement can be made. The *locality ratio* of a local search algorithm is $\min f(S)/f(O)$, where S is a solution that is locally-optimal with respect to the small changes considered by the algorithm, O is a global optimum, and f is the objective function. The locality ratio provides a natural, worst-case guarantee on the approximation performance of the local search algorithm.

In many cases, oblivious local search may have a very poor locality ratio, implying that a locally-optimal solution may be of significantly lower quality than the global optimum. For example, for submodular matroid maximization, the locality ratio for an algorithm changing a single element at each step is $1/2$ [15]. *Non-oblivious* local search attempts to avoid this problem by making use of a secondary potential function to guide the search. By carefully choosing this auxiliary function, we ensure that poor local optima with respect to the original objective function are no longer local optima with respect to the new potential function.

In previous work [14], we designed an optimal non-oblivious local search algorithm for the

¹The basic reason behind this dependence on u is that the fractional solution in general has full support. Our intermediate solutions, being bona fide bases, always have support of size n .

restricted case of maximum coverage subject to a matroid constraint. In this problem, we are given a weighted universe of elements, a collection of sets, and a matroid defined on this collection. The goal is to find a collection of sets that is independent in the matroid and covers elements of maximum total weight. The non-oblivious potential function used in [14] gives extra weight to solutions that cover elements multiple times. In the present work, we extend this approach to general monotone submodular functions. This presents two challenges: defining a non-oblivious potential function without reference to the coverage representation, and analyzing the resulting algorithm.

In order to define the general potential function, we construct a variant of the potential function from [14] which doesn't refer to elements. Instead, the potential function aggregates the information obtained by applying the objective function to all subsets of the input, weighted according to their size. Intuitively, the resulting potential function gives extra weight to solutions that contain a large number of good sub-solutions, or equivalently, remain good solutions on average even when elements are randomly removed. An appropriate setting of the weights defining our potential function yields a function which coincides with the previous definition for coverage functions, but still makes sense for arbitrary monotone submodular functions.

The analysis of the algorithm in [14] is relatively straightforward. For each type of element in the universe of the coverage problem, we must prove a certain inequality among the coefficients defining the potential function. In the general setting, however, we need to construct a proof using only the inequalities given by monotonicity and submodularity. The resulting proof is non-obvious and delicate. For the proof to work, a certain sequence defined by a recurrence relation needs to be non-decreasing. The locality gap can then be read off the sequence. We describe a way to construct the sequence using the recurrence relation in such a way that it is non-decreasing. In order to show that the resulting performance ratio is at least $1 - 1/e$, we use an explicit formula for the sequence in terms of Padé approximants of e^x .

1.2 Related work

Fisher, Nemhauser and Wolsey [25, 15] analyze greedy and local search algorithms for submodular maximization subject to various constraints, including single and multiple matroid constraints, and obtain some of the earliest results in the area, including a $1/(k+1)$ -approximation for monotone submodular maximization subject to k matroid constraints. A recent survey by Goundan and Schulz [17] reviews many results pertaining to the greedy algorithm for submodular maximization.

More recently, Lee, Sviridenko and Vondrák [23] consider the problem of both monotone and non-monotone submodular maximization subject to multiple matroid constraints, attaining a $1/(k+\epsilon)$ -approximation for monotone submodular maximization subject to $k \geq 2$ constraints using local search. Feldman et al. [13] show that a local search algorithm attains the same bound for the related class of k -exchange systems, which includes the intersection of k strongly base orderable matroids, as well as the independent set problem in $(k+1)$ -claw free graphs. Further work by Ward [30] shows that a non-oblivious local search routine attains an improved $2/(k+3) - \epsilon$ approximation for this class of problems.

In the case of unconstrained non-monotone maximization, Feige, Mirrokni and Vondrák [10] give a $2/5$ approximation via a randomized local search algorithm, and give an upper bound of $1/2$ in the value oracle model. Gharan and Vondrák [16] improved the algorithmic result to 0.41 by enhancing the local search algorithm with ideas borrowed from simulated annealing. Feldman, Naor and Schwarz [12] later improved this to 0.42 by using a variant of the continuous greedy algorithm.

Buchbinder, Feldman, Naor and Schwarz have recently obtained an optimal $1/2$ approximation algorithm [4].

In the setting of constrained non-monotone submodular maximization, Lee et al. [22] give a $1/(k+2+1/k+\epsilon)$ approximation subject to k matroid constraints and a $1/(5-\epsilon)$ approximation for k knapsack constraints. Further work by Lee, Sviridenko and Vondrák [23] improves the approximation ratio in the case of k matroid constraints to $1/(k+1+1/(k-1)+\epsilon)$. Feldman et al. [13] attain this ratio for k -exchange systems. In the case of non-monotone submodular maximization subject to a *single* matroid constraint, Feldman, Naor and Schwarz [11] obtain a $1/e$ approximation by using a version of the continuous greedy algorithm. They additionally unify various applications of the continuous greedy and obtain improved approximations for non-monotone submodular maximization subject to a matroid constraint or $O(1)$ knapsack constraints.

1.3 Organization of the paper

The rest of the paper is organized as follows. In Section 2 we give the definition of our algorithm, and present a high-level overview of the proofs and concepts used in its analysis. Section 3 defines the non-oblivious potential functions g and \tilde{g} used by our algorithm and provides some of their properties. Section 4 determines the locality gap and runtime of the resulting algorithm, together with some improvements on it. Appendix A provides some intuition for the definition of g by showing that g agrees with the non-oblivious potential function defined in [14] when $c = 1$. Appendix B gives a detailed analysis of the running time of the continuous greedy algorithm for the sake of comparison.² The rest of the appendices contain proofs of various results claimed in the main body of the paper.

2 The algorithm

Our non-oblivious local search algorithm is shown in Algorithm 1. The input to the algorithm is a matroid \mathcal{M} , a monotone submodular function f , an upper bound on its restricted curvature c (we assume $0 < c \leq 1$), an error parameter ϵ_0 , and a sampling parameter N . The matroid \mathcal{M} is given as a universe \mathcal{U} and a collection of independent sets $\mathcal{I} \subseteq 2^{\mathcal{U}}$, itself given as an independence oracle (an oracle deciding whether $S \in \mathcal{I}$ for an arbitrary subset $S \subseteq \mathcal{U}$). Throughout the paper, we let n denote the rank of \mathcal{M} and $u = |\mathcal{U}|$. The reader interested only in monotone submodular maximization, without any restriction on the curvature, can substitute 1 for c in all that follows.

In Theorem 4.10 we show how to choose ϵ_0 and N in order to obtain (with high probability) an approximation ratio as close to $(1 - e^{-c})/c$ as desired. Theorem 4.12 shows how to use the algorithm as a black-box in order to remove the small extra factor ϵ from the approximation ratio to yield a clean $(1 - e^{-c})/c$ -approximation algorithm, given c . Theorem 4.13 shows how to use the algorithm as a black-box to yield a $((1 - e^{-c})/c - \epsilon)$ -approximation algorithm even without knowing c in advance.

We define the function g in Section 3. The function is defined by a sequence of coefficients $\beta_k^{(m)}$ obtained from an auxiliary sequence $\gamma_k^{(m)}$, depending on the rank of the given matroid. Because our

²Because the rounding part of the continuous greedy algorithm relies on several components, including submodular minimization (pipage rounding) or an efficient implementation of Carathéodory's theorem in the matroid base polytope (swap rounding), it is difficult to give a general runtime for it. We have tried to be as generous as possible in our analysis.

non-oblivious potential function g requires an exponential number of value queries to f to compute exactly, Algorithm 1 makes use of an approximation \tilde{g} to g , obtained by evaluating f taking N samples. We define the random process and sampling used to compute \tilde{g} in Lemma 3.3.

Input: $\mathcal{M} = (\mathcal{U}, \mathcal{I})$, f , c , ϵ_0 , N
Determine the rank of \mathcal{M} and calculate coefficients for g ;
Let \tilde{g} be the approximation to g obtained by using N random samples;
Let S_{init} be the result of running the standard greedy algorithm on (\mathcal{M}, \tilde{g}) ;
 $S \leftarrow S_{\text{init}}$;
repeat
 foreach element $e \in S$ and $x \in \mathcal{U} \setminus S$ **do**
 $S' \leftarrow S \setminus \{e\} \cup \{x\}$;
 if $S' \in \mathcal{I}$ and $\tilde{g}(S') > (1 + \epsilon_0)\tilde{g}(S)$ **then**
 $S \leftarrow S'$;
 break;
until No exchange is made;
return S ;

Algorithm 1: The non-oblivious local search algorithm

The purpose of the greedy phase is to produce a set S_{init} with non-negligible $g(S_{\text{init}})$. This will allow us to bound the number of iterations in the main loop. Instead of running the greedy phase with the function g , we could also run it with f , obtaining marginally inferior results. An even simpler approach is to “guess” a set S_1 such that $g(\{S_1\}) \geq \max g(S)/n$, at a multiplicative $O(n \log n)$ cost in the runtime.

3 The non-oblivious potential function

We now define our non-oblivious potential function g , which will depend on the value of the parameter c . As in the coverage case, our goal is to somehow give extra weight to solutions that will have more flexibility in future iterations of the local search procedure. One way to do this is to incorporate the value of all subsets of a solution S into the value of $g(S)$. Then, we can give some extra value to solutions that contain a large number of good sub-solutions.

With this approach in mind, we define $g(S)$ to be a combination of the values $f(T)$ for all $T \subseteq S$. The extra weight that a subset T contributes will depend on both its size and the size of the solution S on which we are evaluating g . Our function g has the general form

$$g(S) = \sum_{k=1}^{|S|} \sum_{T \in \binom{S}{k}} \frac{\beta_k^{(|S|)}}{\binom{|S|}{k}} f(T) = \sum_{k=1}^{|S|} \beta_k^{(|S|)} \mathbb{E}_{T \in \binom{S}{k}} f(T). \quad (1)$$

Here, $\frac{\beta_k^{(|S|)}}{\binom{|S|}{k}} \geq 0$ is the weight given to the value $f(T)$ on any subset T of size k . Alternatively, we can think of g in terms of the following random process. First, we choose a value k between 1 and $|S|$ with probability proportional to $\beta_k^{(|S|)}$. Then, we choose k items randomly from S to obtain a subset T . The value of g is then (up to a multiplicative constant) precisely the expected value of f on the resulting random subset T .

Note that the coefficients β depend on the size of $|S|$ —that is, we have a separate sequence of coefficients $\beta_1^{(m)}, \dots, \beta_m^{(m)}$ for each value of m (where m corresponds to the size of the set S on which g is being evaluated). The local search phase of Algorithm 1 as well as the proof of Theorem 4.6, which bounds the locality gap of g , requires only that g be defined for sets of size n . The analysis of the initial greedy phase requires that g be defined for sets of size 0 to $2n$, and be monotone submodular. In order to show that g is monotone submodular, and to relate g to the potential function obtained in [14], we require a sequence of coefficients for each m .

In order to complete our definition of the non-oblivious local search algorithm, we must “only” specify appropriate values for these coefficients. We now turn to this task. In later proofs, it will be more convenient to work with expressions of the form $\gamma_\ell^{(m)} = \ell \beta_\ell^{(m)}$. As we have noted, in the execution and analysis of Algorithm 1 we need only consider sets of size at most $2n$ (the initial greedy phase considers sets of size at most $|O \cup S|$). Although they are not needed by the algorithm, our analysis will make use of additional coefficients $\gamma_0^{(m)}$ and $\gamma_{m+1}^{(m)}$ for each value of $m \geq 0$, and additional sequences γ^m for $m > 2n$.

We now give a formal construction for the necessary coefficient sequences. Let $q = 2 \lfloor \frac{n-1}{2} \rfloor + 1$. We set $\gamma_q^{(2n)} = \gamma_{q+1}^{(2n)} = 1$ and define the rest of the sequence $\gamma^{(2n)} = \gamma_0^{(2n)}, \dots, \gamma_{2n+1}^{(2n)}$ using the recurrence

$$\ell \gamma_{\ell+1}^{(m)} = (2\ell - m + c - 1) \gamma_\ell^{(m)} + (m - \ell + 1) \gamma_{\ell-1}^{(m)}, \quad 1 \leq \ell \leq m \quad (\gamma\text{-REC})$$

with $m = 2n$. We normalize the resulting sequence by dividing it by $\gamma_0^{(2n)}$. Note that the normalized sequence still satisfies recurrence (γ -REC), but additionally has $\gamma_0^{(2n)} = 1$. Then, we generate the sequences $\gamma^{(m)}$ for all $1 \leq m < 2n$ and $m > 2n$ from the resulting sequence by repeated use of the downward and upward recurrences

$$\gamma_\ell^{(m-1)} = \gamma_0^{(m)} + \frac{c}{m} \sum_{k=1}^{\ell} \gamma_k^{(m)} \quad (\gamma\text{-DOWN})$$

$$\gamma_{\ell+1}^{(m)} = c^{-1} m \left(\gamma_{\ell+1}^{(m-1)} - \gamma_\ell^{(m-1)} \right), \quad \gamma_0^{(m)} = \gamma_0^{(m-1)}, \quad \gamma_{m+1}^{(m)} = \gamma_m^{(m-1)} \quad (\gamma\text{-UP})$$

Lemma 3.1. *The sequences $\gamma^{(m)} = \gamma_0^{(m)}, \dots, \gamma_{m+1}^{(m)}$, satisfy the following properties:*

- (a) $\gamma^{(m)}$ satisfies recurrence (γ -REC) for all $m \geq 1$.
- (b) $\gamma^{(m)}$ is non-decreasing and non-negative for $0 \leq m \leq 2n$.
- (c) $\gamma^{(m-1)}$ and $\gamma^{(m)}$ satisfy both (γ -DOWN) and (γ -UP) for all $m \geq 1$.

We defer the proof of the lemma to Appendix C. We comment, without proof, that $\gamma_k^{(m)} = \exp \frac{ck}{m+1} + O(c/m)$.

Many of our proofs will require bounding the value of the largest term $\gamma_{m+1}^{(m)}$ in the resulting sequences. According to the last case of (γ -UP), the sequences for $0 \leq m \leq 2n$ all have the same value for this term, which is thus some constant E independent of m . In order to do this, we derive an explicit formula for each sequence:

Lemma 3.2. For all $m \geq 0$, the terms of $\gamma^{(m)}$ are given by the following formula:

$$\gamma_{\ell+1}^{(m)} = (-1)^\ell \sum_{k=0}^{m-1} \frac{m!}{k!} c^{k-m} \left[(-1)^k \binom{m-1-k}{\ell-k} E - \binom{m-1-k}{\ell} \right] \quad (\gamma\text{-CLOSED})$$

where $E = \gamma_{m+1}^{(m)}$ and $0 \leq \ell \leq m-1$.

Using the resulting sequences to set $\beta_k^{(m)} = \gamma_k^{(m)}/k$ in (1), we complete the definition of our potential function. We now consider some of the properties of the resulting function g . As we show in Appendix D, if f is monotone submodular, g is monotone submodular when restricted to sets of size at most $2n$. Moreover, g has slightly lower curvature than f . Unfortunately, evaluating $g(S)$ requires evaluating f on all subsets of S . Hence, we cannot compute g directly without using an exponential number of calls to the value oracle f . In our original description of g , we provided some intuition in terms of a random process. We now formalize this intuition to show that g can be approximated by sampling.

Suppose that $|S| = m \leq 2n$, and define $\alpha_m = \sum_{k=1}^m \beta_k^{(m)}$. We define the random set X using the following two step experiment. First, let L be a random variable taking value k with probability $\beta_k^{(m)}/\alpha_m$, then choose X as a uniformly random subset of S of size L . Then, from the linearity of expectations we have $\alpha_m \mathbb{E} f(X) = g(S)$. We now estimate the error incurred when g is estimated by taking N samples. The following lemma is proved in Appendix E.

Lemma 3.3. Let S be a set of size m . Let N be a positive integer, and X_1, \dots, X_N be N i.i.d. random samples drawn from the distribution for X . Define $\tilde{g} = \frac{1}{N} \sum_{i=1}^N \alpha_m f(X_i)$. For every $\epsilon > 0$,

$$\Pr[|\tilde{g}(S) - g(S)| > \epsilon g(S)] \leq 2 \exp\left(-\frac{\epsilon^2 N}{32 \log^2 m}\right).$$

In order to obtain a polynomial time algorithm, we use \tilde{g} in Algorithm 1 rather than g , where the number of samples N is a parameter to the algorithm.

4 Analysis of the algorithm

In this section, we derive a bound on the approximation ratio and running time of Algorithm 1. Subsection 4.1 calculates (in Theorem 4.6) the locality gap of the local search phase in terms of the γ sequences used to define g . Subsection 4.2 deduces (in Theorem 4.10) the approximation ratio of Algorithm 1, and estimates its running time. Subsection 4.3 shows (in Theorems 4.12 and 4.13) how to obtain a clean $(1 - e^{-c})/c$ approximation (given c), and how to obtain a $(1 - e^{-c})/c - \epsilon$ approximation without knowing c in advance.

4.1 Locality gap

First, we establish some basic terminology used in this section.

Definition 4.1 (Global and Approximate Local Optima). Let $(\mathcal{M} = (\mathcal{U}, \mathcal{I}), f)$ be an instance of monotone submodular maximization over a matroid, where \mathcal{U} is the ground set, \mathcal{I} is the family of independent sets of a matroid, and f is a (normalized) monotone submodular function on $2^{\mathcal{U}}$.

An independent set O is a *global optimum* or *optimal solution* if for every $A \in \mathcal{I}$, $f(O) \geq f(A)$. Since f is monotone, there is always an optimal solution which is a base (maximal independent set) of the matroid. We henceforth consider only global optima that are bases.

Let $\epsilon \geq 0$. A base S is an ϵ -approximate local optimum if for every $a \in S$ and for every $b \in \mathcal{U}$ such that $S \setminus \{a\} \cup \{b\} \in \mathcal{I}$,

$$(1 + \epsilon)g(S) \geq g(S \setminus \{a\} \cup \{b\}).$$

Consider an instance (\mathcal{M}, f) with optimal solution O (without loss of generality, a base), and let S be the solution produced by the algorithm. Below, in Subsection 4.2, we show that S is an ϵ_1 -approximate local optimum of g , where $\epsilon_1 = O(\epsilon_0)$. Our bound, stated in Theorem 4.6, uses the approximate local optimality of $g(S)$ to bound the quality of $f(S)$ in relation to $f(O)$.

Since both O and S are bases, a theorem of Brualdi [3] shows that there exists a bijection $\pi: S \rightarrow O$ such that $S \setminus \{x\} \cup \{\pi(x)\}$ is a base of \mathcal{M} for all $x \in S$. We index the elements of $S = \{s_1, \dots, s_n\}$ arbitrarily, and then for each element $s_i \in S$, we define $o_i = \pi(s_i)$. For a set of indices $I \subseteq [n]$, we use the notation S_I (respectively, O_I) to denote the set $\{s_i : i \in I\}$ (respectively, $\{o_i : i \in I\}$). The notation $[n]$ itself is shorthand for $\{1, \dots, n\}$. Note that this bijection, used here to index the sets of O and S , is essentially the only property of matroids that we require for the remainder of our analysis.

It will be convenient to work with the following symmetric notation. Let l, b, g be non-negative integers satisfying $l + b \leq n$ and $g + b \leq n$. Then, we define $X_{l,b,g}$ to be the multiset of sets $(S_L \cup O_G)$ for all distinct L, G , satisfying $|L| = l + b$, $|G| = g + b$, $|L \cap G| = b$. That is, $X_{l,b,g}$ is the collection of all sets containing $l + b$ elements from S , and $g + b$ elements from O , where b of the elements have the same index in both S and O . Note that if we have some element x in $S \cap O$, then sets containing x will appear multiple times in $X_{l,b,g}$, once when x is treated as an element of S and once when it is treated as an element of O . Hence, we have $|X_{l,b,g}| = \binom{n}{l} \binom{n-l}{g} \binom{n-l-g}{b}$. We define $F_{l,b,g}$ to be the expected value of a uniformly random set in $X_{l,b,g}$:

$$F_{l,b,g} = \frac{1}{|X_{l,b,g}|} \sum_{a \in X_{l,b,g}} f(a).$$

We adopt the convention that $F_{l,b,g} = 0$ if one of l, b, g is negative.

The proof of Theorem 4.6 makes use of several inequalities following from local optimality, the definition of g and the submodularity of f . Our first ancillary inequality simply re-expresses the (approximate) local optimality of S in our symmetric notation:

Lemma 4.2.

$$\epsilon_1 n \sum_{k=1}^n \beta_k^{(n)} F_{k,0,0} + \sum_{k=1}^n \gamma_k^{(n)} (F_{k,0,0} - F_{k-1,0,1}) \geq 0.$$

Proof. Note that $S \setminus \{s_i\} \cup \{o_i\}$ is a base for all $1 \leq i \leq n$. Since S is an ϵ_1 -approximate local optimum, we must have $(1 + \epsilon_1)g(S) \geq g(S \setminus \{s_i\} \cup \{o_i\})$ for all such i . Summing over $1 \leq i \leq n$ gives:

$$\epsilon_1 n g(S) + n g(S) - \sum_{i=1}^n g(S \setminus \{s_i\} \cup \{o_i\}) \geq 0. \quad (2)$$

From the definition of g and F , we have $g(S) = \sum_{k=1}^n \beta_k^{(n)} F_{k,0,0}$. We now focus on the final summation in inequality (2). Consider an arbitrary set in $X_{k,0,0}$. This set has the form S_I where

$|I| = k$, and appears as a subset of $S \setminus \{s_i\} \cup \{o_i\}$ for each value of $i \notin I$. Thus, it appears in the sum $(n - k)$ times, each with weight $\frac{\beta_k^{(n)}}{\binom{n}{k}} = \frac{\beta_k^{(n)}}{|X_{k,0,0}|}$. Hence, the coefficient of $F_{k,0,0}$ in the sum is $(n - k)\beta_k^{(n)}$. Now, consider an arbitrary set in $X_{k-1,0,1}$. This set has the form $S_I \cup O_i$ for some I with $|I| = k - 1$ and $i \notin I$. Each such set appears as a subset of $S \setminus \{s_i\} \cup \{o_i\}$ for exactly one value of i and so appears in the sum once, with weight $\frac{\beta_k^{(n)}}{\binom{n}{k}} = \frac{k\beta_k^{(n)}}{n\binom{n-1}{k-1}} = \frac{k\beta_k^{(n)}}{|X_{k-1,0,1}|}$. Hence, the coefficient of $F_{k-1,0,1}$ in the sum is $k\beta_k^{(n)}$. No other sets appear in the sum. It follows that the final summation is equivalent to

$$\sum_{k=1}^n (n - k)\beta_k^{(n)} F_{k,0,0} + k\beta_k^{(n)} F_{k-1,0,1},$$

and the claim follows from the definition $\gamma_k^{(n)} \triangleq k\beta_k^{(n)}$. \square

Our remaining lemma follow directly from the monotonicity and submodularity of f .

Lemma 4.3. *For ℓ satisfying $0 \leq \ell \leq n$,*

$$(n - \ell)F_{\ell,0,1} + \ell F_{\ell-1,0,1} \geq \ell F_{\ell-1,0,0} + (n - \ell - c)F_{\ell,0,0} + f(O).$$

In order to prove Lemma 4.3, we first prove two smaller inequalities following from submodularity and monotonicity of f .

Lemma 4.4. *For ℓ satisfying $0 \leq \ell \leq n$, $(n - \ell)F_{\ell,0,1} \geq (n - \ell - 1)F_{\ell,0,0} + F_{\ell,0,n-\ell}$.*

Proof. When $\ell = n$, the inequality reads $0 \geq -F_{n,0,0} + F_{n,0,0}$. For $\ell = n - 1$, it reads $F_{n-1,0,1} \geq F_{n-1,0,1}$. So suppose $\ell \leq n - 2$.

We show by induction that for $\ell + 1 \leq k \leq n$,

$$\sum_{i=\ell+1}^k f(S_{[\ell]} \cup O_{\{i\}}) \geq f(S_{[\ell]} \cup O_{\{\ell+1, \dots, k\}}) + (k - \ell - 1)f(S_{[\ell]}).$$

The case $k = \ell + 1$ is trivial. For the induction step, we have:

$$\begin{aligned} \sum_{i=\ell+1}^{k+1} f(S_{[\ell]} \cup O_{\{i\}}) &\geq f(S_{[\ell]} \cup O_{\{k+1\}}) + f(S_{[\ell]} \cup O_{\{\ell+1, \dots, k\}}) + (k - \ell - 1)f(S_{[\ell]}) \\ &\geq f(S_{[\ell]} \cup O_{\{\ell+1, \dots, k+1\}}) + f(S_{[\ell]}) + (k - \ell - 1)f(S_{[\ell]}), \end{aligned}$$

where the first inequality follows from the induction hypothesis, and the second from submodularity of f . Taking $k = n$ and then averaging over all permutations of indices yields the required inequality. \square

Lemma 4.5. *For ℓ satisfying $0 \leq \ell \leq n$, $F_{\ell,0,n-\ell} + \ell F_{\ell-1,0,1} \geq \ell F_{\ell-1,0,0} + (1 - c)F_{\ell,0,0} + f(O)$.*

Proof. When $\ell = 0$, the inequality reads $F_{0,0,n} \geq (1 - c)F_{0,0,0} + f(O)$, which is true since $F_{0,0,n} = f(O)$ and $F_{0,0,0} = f(\emptyset) = 0$. So suppose $\ell \geq 1$.

Let $C = \{i \in [\ell] : s_i = o_i\}$. We show by induction that for $0 \leq k \leq \ell$,

$$\begin{aligned} f(S_{[\ell]} \cup O_{\{\ell+1, \dots, n\}}) + \sum_{i=1}^k f(S_{[\ell] \setminus \{i\}} \cup O_{\{i\}}) \\ \geq f(S_{[\ell]} \cup O_{[k] \cup \{\ell+1, \dots, n\}}) + \sum_{i=1}^k f(S_{[\ell] \setminus \{i\}}) + (1-c) \sum_{i \in C \cap [k]} f(\{s_i\}). \end{aligned} \quad (3)$$

The case $k = 0$ is trivial. For the induction step, there are two cases. If $s_i \neq o_i$ then $i \notin C$. In this case,

$$\begin{aligned} f(S_{[\ell]} \cup O_{\{\ell+1, \dots, n\}}) + \sum_{i=1}^{k+1} f(S_{[\ell] \setminus \{i\}} \cup O_{\{i\}}) \\ \geq f(S_{[\ell] \setminus \{k+1\}} \cup O_{\{k+1\}}) + f(S_{[\ell]} \cup O_{[k] \cup \{\ell+1, \dots, n\}}) + \sum_{i=1}^k f(S_{[\ell] \setminus \{i\}}) + (1-c) \sum_{i \in C \cap [k]} f(\{s_i\}) \\ \geq f(S_{[\ell]} \cup O_{[k+1] \cup \{\ell+1, \dots, n\}}) + f(S_{[\ell] \setminus \{k+1\}}) + \sum_{i=1}^k f(S_{[\ell] \setminus \{i\}}) + (1-c) \sum_{i \in C \cap [k]} f(\{s_i\}), \end{aligned}$$

where the first inequality follows from the induction hypothesis, and the second from submodularity of f .

If $s_i = o_i$ then $i \in C$. In this case,

$$\begin{aligned} f(S_{[\ell]} \cup O_{\{\ell+1, \dots, n\}}) + \sum_{i=1}^{k+1} f(S_{[\ell] \setminus \{i\}} \cup O_{\{i\}}) \\ \geq f(S_{[\ell] \setminus \{k+1\}} \cup O_{\{k+1\}}) + f(S_{[\ell]} \cup O_{[k] \cup \{\ell+1, \dots, n\}}) + \sum_{i=1}^k f(S_{[\ell] \setminus \{i\}}) + (1-c) \sum_{i \in C \cap [k]} f(\{s_i\}) \\ \geq f(S_{[\ell]} \cup O_{[k+1] \cup \{\ell+1, \dots, n\}}) + f(S_{[\ell]}) + \sum_{i=1}^k f(S_{[\ell] \setminus \{i\}}) + (1-c) \sum_{i \in C \cap [k]} f(\{s_i\}) \\ \geq f(S_{[\ell]} \cup O_{[k+1] \cup \{\ell+1, \dots, n\}}) + f(S_{[\ell] \setminus \{k+1\}}) + (1-c) f(\{s_{k+1}\}) + \sum_{i=1}^k f(S_{[\ell] \setminus \{i\}}) + (1-c) \sum_{i \in C \cap [k]} f(\{s_i\}), \end{aligned}$$

where the first inequality follows from the induction hypothesis, the second from submodularity of f and the last from curvature of f .

Taking $k = \ell$ in (3), we have:

$$\begin{aligned}
& f(S_{[\ell]} \cup O_{\{\ell+1, \dots, n\}}) + \sum_{i=1}^{\ell} f(S_{[\ell] \setminus \{i\}} \cup O_{\{i\}}) \\
& \geq f(S_{[\ell]} \cup O_{[\ell] \cup \{\ell+1, \dots, n\}}) + \sum_{i=1}^{\ell} f(S_{[\ell] \setminus \{i\}}) + (1-c) \sum_{i \in C} f(\{s_i\}) \\
& \geq f(O_{[\ell] \cup \{\ell+1, \dots, n\}}) + (1-c) f(S_{[\ell] \setminus C}) + \sum_{i=1}^{\ell} f(S_{[\ell] \setminus \{i\}}) + (1-c) \sum_{i \in C} f(\{s_i\}) \\
& \geq f(O) + (1-c) f(S_{[\ell]}) + \sum_{i=1}^{\ell} f(S_{[\ell] \setminus \{i\}}),
\end{aligned}$$

where the second inequality follows from curvature of f and the final line follows from submodularity of f . Averaging over all permutations of indices then gives the claimed inequality. \square

Proof of Lemma 4.3. Adding the inequalities from Lemmas 4.4 and 4.5 gives

$$(n - \ell)F_{\ell,0,1} + \ell F_{\ell-1,0,1} \geq \ell F_{\ell-1,0,0} + (n - \ell - c)F_{\ell,0,0} + f(O). \quad \square$$

Theorem 4.6. Suppose O is a global optimum of f and that S is an ϵ_1 -approximate local optimum of g . Then,

$$(1 + 8\epsilon_1 n \log n) \gamma_{n+1}^{(n)} f(S) \geq c^{-1} (\gamma_{n+1}^{(n)} - \gamma_0^{(n)}) f(O).$$

Proof. Consider the inequality given in Lemma 4.2. From monotonicity of f , we have $F_{k,0,0} \leq F_{n,0,0}$ for all $k \leq n$. Using Lemma E.2 from Appendix E, which gives an upper bound on the value $\alpha_n = \sum_{k=1}^n \beta_n^{(k)}$, we obtain:

$$\epsilon_1 n \sum_{k=1}^n \beta_k^{(n)} F_{k,0,0} \leq \epsilon_1 n \sum_{k=1}^n \beta_k^{(n)} F_{n,0,0} \leq 8\epsilon_1 n \log n F_{n,0,0} \leq 8\epsilon_1 \gamma_{n+1}^{(n)} n \log n F_{n,0,0}$$

where in the last line we have used the fact that $\gamma_n^{(n+1)} \geq \gamma_1^{(n+1)} = 1$ from Lemma 3.1. Furthermore, since $F_{0,0,0} = f(\emptyset) = 0$ and $F_{-1,0,1} = 0$, we have $\gamma_0^{(n)} (F_{0,0,0} - F_{-1,0,0}) = 0$, and so the inequality in Lemma 4.2 gives

$$8\epsilon_1 \gamma_{n+1}^{(n)} n \log n F_{n,0,0} + \sum_{k=0}^n \gamma_k^{(n)} (F_{k,0,0} - F_{k-1,0,1}) \geq 0. \quad (4)$$

Since by Lemma 3.1 part (b), $\gamma^{(n)}$ is non-decreasing, we have $c^{-1}(\gamma_{\ell+1}^{(n)} - \gamma_{\ell}^{(n)}) \geq 0$ for all $0 \leq \ell \leq n$, and multiplying the inequality from Lemma 4.3 by $c^{-1}(\gamma_{\ell+1}^{(n)} - \gamma_{\ell}^{(n)})$ gives

$$c^{-1}(\gamma_{\ell+1}^{(n)} - \gamma_{\ell}^{(n)}) [(n - \ell)F_{\ell,0,1} + \ell F_{\ell-1,0,1} - \ell F_{\ell-1,0,0} - (n - \ell - c)F_{\ell,0,0}] \geq c^{-1}(\gamma_{\ell+1}^{(n)} - \gamma_{\ell}^{(n)}) f(O), \quad (5)$$

for $\ell \in \{0, \dots, n\}$. We claim that the inequality that results from adding the $n + 1$ inequalities given by (5) to inequality (4) is the desired inequality.

We first consider all terms of the form $F_{k,0,0}$. For $0 \leq k \leq n-1$, the coefficient of $F_{k,0,0}$ is

$$\begin{aligned} \gamma_k^{(n)} - c^{-1}(n-k-c)(\gamma_{k+1}^{(n)} - \gamma_k^{(n)}) - c^{-1}(k+1)(\gamma_{k+2}^{(n)} - \gamma_{k+1}^{(n)}) \\ = c^{-1}(n-k)\gamma_k^{(n)} + c^{-1}(2k-n+c+1)\gamma_{k+1}^{(n)} - c^{-1}(k+1)\gamma_{k+2}^{(n)} = 0, \end{aligned}$$

where the final equality follows from $(\gamma\text{-REC})$. The coefficient of $F_{n,0,0} = f(S)$ is

$$8\epsilon_1\gamma_{n+1}^{(n)}n \log n + \gamma_n^{(n)} + c^{-1}c(\gamma_{n+1}^{(n)} - \gamma_n^{(n)}) = (1 + 8\epsilon_1n \log n)\gamma_{n+1}^{(n)}.$$

Now, we consider all terms of the form $F_{k-1,0,1}$. For $1 \leq k \leq n$, the coefficient of $F_{k-1,0,1}$ is

$$\begin{aligned} -\gamma_k^{(n)} + c^{-1}(n-k+1)(\gamma_k^{(n)} - \gamma_{k-1}^{(n)}) + c^{-1}k(\gamma_{k+1}^{(n)} - \gamma_k^{(n)}) = \\ c^{-1}k\gamma_{k+1}^{(n)} - c^{-1}(2k-n+c-1)\gamma_k^{(n)} - c^{-1}(n-k+1)\gamma_{k-1}^{(n)} = 0, \end{aligned}$$

where the final equality follows from $(\gamma\text{-REC})$. Additionally, $F_{-1,0,1} = 0$ by definition. Finally, the coefficient of $f(O)$ is

$$\sum_{k=0}^n c^{-1}(\gamma_{k+1}^{(n)} - \gamma_k^{(n)}) = c^{-1}(\gamma_{n+1}^{(n)} - \gamma_0^{(n)}). \quad \square$$

4.2 Approximation ratio

Now, we must translate the bounds that Theorem 4.6 gives for the locality gap into a performance guarantee for Algorithm 1. We already have $\gamma_0^{(n)} = 1$ by Lemma 3.1 (c). We now bound $\gamma_{n+1}^{(n)}$. In order to do this, we make use of a surprising connection between the explicit formula $(\gamma\text{-CLOSED})$ and the Padé approximants to the function e^x , defined in [26, §66]. For $\mu, \nu \geq 0$, the (μ, ν) -Padé approximant to e^x is given by $R_{\mu,\nu} = \frac{P_{\mu,\nu}}{Q_{\mu,\nu}}$, whose numerator and denominator are defined by:

$$P_{\mu,\nu}(x) = \sum_{k=0}^{\mu} \frac{x^k(\mu+\nu-k)! \mu!}{(\mu+\nu)! k! (\mu-k)!}, \quad Q_{\mu,\nu}(x) = \sum_{k=0}^{\nu} \frac{(-x)^k(\mu+\nu-k)! \nu!}{(\mu+\nu)! k! (\nu-k)!}.$$

Both $P_{\mu,\nu}(x)$ and $Q_{\mu,\nu}(x)$ are positive (for the latter, note that each term in the alternating sum is of smaller magnitude than the preceding one). Additionally, we have the following formula from [26, §66]:

$$R_{\mu,\nu}(x) = e^x + (-1)^{\nu+1} x^{\mu+\nu+1} \frac{\int_0^1 e^{xt} t^{\nu} (1-t)^{\mu} dt}{(\mu+\nu)! Q_{\mu,\nu}(x)}. \quad (6)$$

We can restate the explicit formula $(\gamma\text{-CLOSED})$ in terms of Padé numerators and denominators as

$$\gamma_{\ell+1}^{(m)} = (-1)^{\ell} m! c^{-m} \binom{m-1}{\ell} [Q_{m-1-\ell,\ell}(c)E - P_{m-1-\ell,\ell}(c)], \quad (7)$$

for all $m \geq 1$, and $1 \leq \ell \leq m$, where $E \triangleq \gamma_{m+1}^{(m)}$. This is proved formally in Lemma C.5, found in Appendix C. Using Equation (6), we derive the following bound on $\gamma_{n+1}^{(n)}$.

Lemma 4.7. $\gamma_{n+1}^{(n)} \geq e^c$.

Proof. By construction, $\gamma_q^{(2n)} = \gamma_{q+1}^{(2n)}$ and so $\gamma_{q+1}^{(2n+1)} = 0$. Thus formula (7) implies that $E = R_{2n-q,q}(c)$. Since q is odd, expression (6) immediately implies that $E > e^c$. \square

As an aside, we note that (6) implies that if we put $E = e^c$ and use (γ -CLOSED) to define $\gamma^{(m)}$, then $\gamma^{(m)}$ is non-decreasing for all m .

Before stating our main theorems, we need the following elementary inequality.

Lemma 4.8. *Let $\delta \leq 1/2$. Suppose that $|\tilde{g}(A) - g(A)| \leq \delta g(A)$, $|\tilde{g}(B) - g(B)| \leq \delta g(B)$ and $(1 + \delta)\tilde{g}(A) \geq \tilde{g}(B)$. Then $(1 + 7\delta)g(A) \geq g(B)$.*

Proof. The premises imply

$$(1 + \delta)^2 g(A) \geq (1 + \delta)\tilde{g}(A) \geq \tilde{g}(B) \geq (1 - \delta)g(B).$$

Therefore

$$\frac{(1 + \delta)^2}{1 - \delta} g(A) \geq g(B).$$

The expression on the left is bounded by

$$\frac{(1 + \delta)^2}{1 - \delta} = 1 + \frac{\delta(3 + \delta)}{1 - \delta} \leq 1 + 7\delta,$$

Since the function $(3 + \delta)/(1 - \delta)$ is increasing and $\delta \leq 1/2$. \square

We also need to know the approximation ratio of the greedy algorithm when the oracle for the function is only approximate. Similar results appear in Goundan and Schulz [17] and Calinescu et al. [6], though their measure of approximation for the oracle is different.

Lemma 4.9. *Let $S_{\text{init}} = \{S_1, \dots, S_n\}$ satisfy*

$$(1 + \eta)g(S_{[k]}) \geq g(S_{[k-1]} \cup \{x\})$$

for all $k \in [n]$ and all $x \in \mathcal{U}$ such that $S_{[k-1]} \cup \{x\} \in \mathcal{I}$. Let g^ be the maximum value taken by g on \mathcal{I} . Then*

$$g(S_{\text{init}}) \geq \frac{g^*}{2 + n\eta}.$$

Proof. During the lemma, we will use the fact that g is monotone and submodular whenever all sets involved are of cardinality at most $2n$.

Suppose $g^* = g(O)$, where $O = \{O_1, \dots, O_n\}$ and $S_{[k-1]} \cup O_{\{k\}} \in \mathcal{I}$ for all $k \in [n]$. Then

$$(1 + \eta) \sum_{k=1}^n g(S_{[k]}) \geq \sum_{k=1}^n g(S_{[k-1]} \cup O_{\{k\}}).$$

Since $g(S_{[k]}) \leq g(S_{\text{init}})$, this implies

$$\begin{aligned}
(1 + n\eta)g(S_{\text{init}}) &= n\eta g(S_{\text{init}}) + \sum_{k=1}^n [g(S_{[k]}) - g(S_{[k-1]})] \\
&\geq \sum_{k=1}^n [g(S_{[k-1]} \cup O_{\{k\}}) - g(S_{[k-1]})] \\
&\geq \sum_{k=1}^n [g(S_{\text{init}} \cup O_{\{k\}}) - g(S_{\text{init}})] \\
&\geq g(S_{\text{init}} \cup O) - g(S_{\text{init}}) \\
&\geq g(O) - g(S_{\text{init}}).
\end{aligned}$$

Rearranging,

$$(2 + n\eta)g(S_{\text{init}}) \geq g(O). \quad \square$$

Now, we are ready to state our main theorems regarding the performance of Algorithm 1.

Theorem 4.10. *Given $0 < \epsilon \leq 1$ and $c \in (0, 1]$, set the parameters of Algorithm 1 as follows:*

$$\epsilon_0 = \frac{\epsilon}{56n \log n}, \quad N = 64\epsilon^{-2}n^2 \log^4 n \log(60\epsilon^{-1}n^2 u \log n).$$

With probability $1 - o(1)$, Algorithm 1 is a $\frac{1-e^{-c}}{c} - \epsilon$ approximation algorithm, running in time $\tilde{O}(\epsilon^{-3}n^4 u)$.

Proof. We analyze the algorithm under the assumption that whenever we evaluate $\tilde{g}(S)$, the value we obtain satisfies

$$(1 - \epsilon_0)g(X) \leq \tilde{g}(X) \leq (1 + \epsilon_0)g(X).$$

Later we will show that this happens with probability $1 - o(1)$.

Let O be the optimal solution for the instance we are considering and let S be the solution produced by the algorithm. Lemma 4.8 shows that S is a $7\epsilon_0$ -approximate local optimum of g . Theorem 4.6 implies that

$$(1 + 56\epsilon_0 n \log n)f(S) \geq \frac{1 - 1/E}{c}f(O).$$

Lemma 4.7 completes the proof of the approximation ratio of the algorithm.

We now bound the number of improvements our algorithm can make. Let g^* be the maximum value taken by g on \mathcal{I} . Applying Lemma 4.9 with $1 + \eta = (1 + \epsilon_0)/(1 - \epsilon_0) \leq 1 + 3\epsilon_0$ (using $\epsilon_0 \leq 1/56$), we deduce (this time using $\epsilon_0 \leq 1/(56n)$)

$$\tilde{g}(S_{\text{init}}) \geq (1 - \epsilon_0)g(S_{\text{init}}) \geq (1 - \epsilon_0)\frac{g^*}{2 + 3n\epsilon_0} \geq \frac{g^*}{2 + 8n\epsilon_0}.$$

Every time the algorithm applies an improvement, it must improve $\tilde{g}(S)$ by at least a factor of $(1 + \epsilon_0)$. Furthermore, $\tilde{g}(S) \leq (1 + \epsilon_0)g^*$ for all S we query. Thus, the number of improvements Algorithm 1 can make is at most:

$$\log_{1+\epsilon_0} \frac{(1 + \epsilon_0)g^*}{\tilde{g}(S_{\text{init}})} \leq \log_{1+\epsilon_0} (1 + \epsilon_0)(2 + 8n\epsilon_0) \leq \log_{1+\epsilon_0} (2 + 11n\epsilon_0) \leq \epsilon_0^{-1} = 56\epsilon^{-1}n \log n.$$

The second inequality follows from the bound $\epsilon_0 \leq 1/(56n)$.

Finally, we derive a bound on the number of samples needed to ensure that with high probability $|\tilde{g}(X) - g(X)| \leq \epsilon_0 g(X)$ for all sets considered by the algorithm. The initial greedy step requires at most nu total evaluations of g , and each improvement step requires at most nu evaluations. Thus, the algorithm requires at most $G = 60\epsilon^{-1}n^2u \log n$ total evaluations of g . Define

$$N = 64\epsilon_0^{-2} \log^2 n \log G = 64\epsilon^{-2} n^2 \log^4 n \log G.$$

Lemma 3.3 shows that the probability that for a given set X , $|\tilde{g}(X) - g(X)| > \epsilon_0 g(X)$, is at most $2/G^2$. Hence this never happens for any set considered by the algorithm with probability at least $1 - 2/G = 1 - o(1)$.

The final algorithm requires a total of $\tilde{O}(\epsilon_0^{-3}n^4u)$ calls to the value oracle for f and $O(\epsilon_0^{-1}n^2u \log n)$ calls to the independence oracle for \mathcal{M} . Its runtime is proportional to the total number of oracle calls to f . \square

4.3 Clean approximation

Algorithm 1 has two shortcomings: it requires prior knowledge of c , and it only approximates the optimal approximation ratio $(1 - e^{-c})/c$. In this subsection we show how to overcome each of these shortcomings individually. Combining the two techniques together, we get a clean $(1 - e^{-c})/c$ approximation (for technical reasons, we need to assume a lower bound on c).

We can remove the ϵ from our approximation ratio by using a partial enumeration technique described by Khuller et al. [21] and employed by Calinescu et al. [5]. Effectively, we try to “guess” a single set in the optimal solution, and then run Algorithm 1 on an instance in which all solutions contain this set. We then iterate over all possible guesses.

Formally, for a matroid $\mathcal{M} = (\mathcal{U}, \mathcal{I})$ and an element $x \in \mathcal{U}$, the contracted matroid \mathcal{M}/x is a matroid on $\mathcal{U} \setminus \{x\}$ in which a set A is independent if and only if $A \cup \{x\} \in \mathcal{I}$. Similarly, for $x \in \mathcal{U}$ we define the contracted function $f_x(A)$ on $\mathcal{U} \setminus \{x\}$ by $f_x(A) = f(A \cup \{x\}) - f(\{x\})$ and note that if f is monotone submodular, then so is f_x .

We can now formulate the new algorithm. Algorithm 2 simply runs Algorithm 1 with suitable parameters on the instance $\mathcal{M}/x, f_x$ for each $x \in \mathcal{F}$, and returns the best resulting solution. The algorithm uses the function

$$\rho(x) = \frac{1 - e^{-x}}{x},$$

which gives the approximation ratio for given curvature.

Input: $\mathcal{M} = (\mathcal{U}, \mathcal{I}), f, c$

Set

$$\epsilon = \frac{1 - \rho(c)}{n}, \quad \epsilon_0 = \frac{\epsilon}{56n \log n}, \quad N = 64\epsilon^{-2}n^2 \log^4 n \log(60\epsilon^{-1}n^2u \log n).$$

for $x \in \mathcal{U}$ **do**

 Let S_x be the result of running Algorithm 1 on $(\mathcal{M}/x, f_x, c, \epsilon_0, N)$;

 Let $y = \operatorname{argmax}_{x \in \mathcal{U}} f(S_x \cup \{x\})$;

return $S_y \cup \{y\}$

Algorithm 2: Clean $(1 - e^{-c})/c$ approximation algorithm

Lemma 4.11. *If Algorithm 1, with parameters as in Algorithm 2, has an approximation ratio of θ on matroids of rank $n - 1$, then Algorithm 2 has an approximation ratio of $1/n + (1 - 1/n)\theta$ on matroids of rank n .*

Proof. Consider an instance (\mathcal{M}, f) , where n is the rank of \mathcal{M} . Let O be some optimal solution for this instance, and $y = \arg \max_{x \in O} f(\{x\})$. Submodularity implies that $f(O) \leq \sum_{x \in O} f(\{x\}) \leq nf(\{y\})$. Furthermore, since Algorithm 1 is a θ -approximation algorithm, we have

$$f_y(S_y) \geq \theta f_y(O \setminus \{y\}) = f(O).$$

Let S be the solution produced by Algorithm 2 on the instance (\mathcal{M}, f) . Then,

$$\begin{aligned} f(S) &= f(\{y\}) + f_y(S_y) \geq f(\{y\}) + \theta f_y(O \setminus \{y\}) = f(\{y\}) + \theta(f(O) - f(\{y\})) \\ &= (1 - \theta)f(\{y\}) + \theta f(O) \geq \left(\frac{1 - \theta}{n} + \theta\right) f(O) = \left(\frac{1}{n} + \frac{n - 1}{n}\theta\right) f(O). \quad \square \end{aligned}$$

Theorem 4.12. *Let $c \in (0, 1]$, and suppose f has restricted curvature c . With probability $1 - o(1)$, Algorithm 2 is a $(1 - e^{-c})/c$ -approximation algorithm running in time $\tilde{O}_c(n^7 u^2)$.*

Proof. Theorem 4.10 implies that each instance of Algorithm 1 run within Algorithm 2 has an approximation ratio of $(1 - e^{-c})/c - \epsilon$, where $\epsilon = (1 - \rho(c))/n$. From Lemma 4.11, Algorithm 2 is then a

$$\frac{1}{n} + \left(1 - \frac{1}{n}\right) \left(\rho(c) - \frac{1 - \rho(c)}{n}\right) = \frac{1}{n^2} + \left(1 - \frac{1}{n^2}\right) \rho(c) \geq \rho(c)$$

approximation. Algorithm 2 calls Algorithm 1 a total of n times with $\epsilon^{-1} = \Theta(n/(1 - \rho(c)))$. Its runtime is hence $\tilde{O}((1 - \rho(c))^{-3} n^8 u)$. \square

Algorithm 1 requires knowledge of c . We can eliminate this need by enumerating over possible values of c with enough granularity. Since the function $\rho(c)$ is continuous, an error in estimating c will only result in a small decrease in the resulting approximation ratio.

Input: $\mathcal{M} = (\mathcal{U}, \mathcal{I})$, f , ϵ

Set

$$\epsilon_2 = \frac{\epsilon}{2}, \quad \epsilon_0 = \frac{\epsilon_2}{56n \log n}, \quad N = 64\epsilon_2^{-2} n^2 \log^4 n \log(60\epsilon_2^{-1} n^2 u \log n).$$

Let $C = \{k\epsilon : 1 \leq k \leq \lfloor 1/\epsilon \rfloor\} \cup \{1\}$;

for $c \in C$ **do**

 Let S_c be the result of running Algorithm 1 on $(\mathcal{M}, f, c, \epsilon_0, N)$;

return $\arg \max_{c \in C} f(S_c)$

Algorithm 3: Algorithm not requiring prior knowledge of the curvature

Theorem 4.13. *Suppose f has curvature c . With probability $1 - o(1)$, Algorithm 3 is a $(1 - e^{-c})/c - \epsilon$ -approximation algorithm running in time $\tilde{O}(\epsilon^{-4} n^4 u)$.*

Proof. Let $c_0 = \min\{x \in C : x \geq c\}$. Clearly $c \leq c_0 < c + \epsilon$. Since f has curvature at most c_0 , Theorem 4.10 shows that $f(S_{c_0})/f(O) \geq \rho(c_0) - \epsilon/2$, where O is a global optimum. Elementary calculus shows that for $x \in [0, 1]$, $-1/2 \leq \rho'(x) < -1/4$. Therefore the approximation ratio of the algorithm is at least $\rho(c_0) - \epsilon/2 > \rho(c + \epsilon) - \epsilon/2 \geq \rho(c) - \epsilon$. Finally, we run Algorithm 1 a total of $\lceil 1/\epsilon \rceil$ times, leading to a running time of $\tilde{O}(\epsilon^{-4} n^4 u)$. \square

The methods of Algorithms 2 and 3 can be put together to obtain a clean $(1 - e^{-c})/c$ -approximation algorithm that doesn't require knowledge of c . The reader who is interested in carrying out the construction will discover that for technical reasons, this approach only works given some lower bound on c .

5 Future work

An immediate open question is whether our algorithm can be made deterministic, even if only for particular classes of functions. If f is given a coverage function, we have already shown [14] that g can be computed explicitly. However, this result required access to the representation of f and so is not possible in the general value oracle model. Even reducing the amount of sampling needed to compute g would be useful, as it would improve the runtime of the algorithm.

A more general question is whether this approach can be extended to other submodular maximization problems, including non-monotone maximization or monotone maximization over multiple matroid constraints. A major difficulty in extending the continuous greedy algorithm to this latter case is that the pipage rounding phase does not generalize to multiple matroids. As our general technique does not require any rounding, it is a natural candidate for improvement in this area.

Finally, we ask whether it is possible to match the improved performance of the continuous greedy algorithm for other problems or restricted settings by combinatorial algorithms similar to our own. Our algorithm as it stands already matches the performance of the continuous greedy algorithm not only in the classical case but also when the objective function has restricted curvature. It is unclear, however, whether our algorithm may match or improve results for other applications of the continuous greedy algorithm, such as those presented in [11].

Appendices

A Coverage functions

In our previous paper [14], we discuss the case where f is a coverage function. In this case, U is a collection of subsets of some other set V . The set V has an associated non-negative weight function w , and the function f is the total weight of elements covered by the sets in its input. The non-oblivious potential function used in [14] is

$$G(S) = \frac{1}{E-1} \sum_{x \in V} \alpha_{h_x(S)} w(x),$$

where $h_x(S) = |A \in S : x \in A|$ is the number of sets containing x , and the sequence α_H is given by

$$\alpha_0 = 0, \quad \alpha_1 = E - 1, \quad \alpha_{H+1} = (H + 1)\alpha_H - H\alpha_{H-1} - 1. \quad (8)$$

In [14] we give a concrete value for E .

Even though the function G is presented quite differently from the function g defined in the present paper, the two functions coincide when $c = 1$.

Lemma A.1. *Suppose that f is a coverage function. For any set S ,*

$$g(S) = (E - 1)G(S).$$

Proof. We will show that

$$g(S) = \sum_{x \in V} \alpha_{h_x(S)} w(x).$$

From the definition of g , it is immediate that there exist constants $\zeta_H^{(m)}$ such that

$$g(S) = \sum_{x \in V} \zeta_{h_x(S)}^{(|S|)} w(x).$$

A priori, the coefficients $\zeta_i^{(m)}$ depend on m . However, our definition of g ensures that this is not the case. Consider the following thought experiment. Add to U an element \emptyset consisting of the empty set. For every $A \subseteq S$ we have $f(A \cup \{\emptyset\}) = f(A)$, and so Lemma D.2 implies that $g(S \cup \{\emptyset\}) = g(S)$. On the other hand,

$$g(S \cup \{\emptyset\}) = \sum_{x \in V} \zeta_{h_x(S)}^{(|S|+1)} w(x).$$

Since S is arbitrary, we must have $\zeta^{(|S|+1)} = \zeta^{(S)}$. Denoting the common value of all these sequences by ζ , we have

$$g(S) = \sum_{x \in V} \zeta_{h_x(S)} w(x).$$

Our task is now reduced to showing $\zeta = \alpha$.

We can get a nice formula for ζ_H by considering the set $S_H = \{A_i : 1 \leq i \leq H\}$, where $A_i = \{x\}$ and x is an element of unit weight. Clearly $g(S_H) = \zeta_H$. On the other hand,

$$g(S_H) = \sum_{k=1}^H \beta_k^{(H)} \mathbb{E}_{T \in \binom{S_H}{k}} f(T) = \sum_{k=1}^H \beta_k^{(H)}.$$

Therefore

$$\zeta_H = \sum_{k=1}^H \beta_k^{(H)}. \quad (9)$$

It remains to deduce recurrence (8) for ζ . The first step is to get another expression for ζ_H . Consider the same set S_H used previously. Taking $S_H \cup \{\emptyset\}$ this time, we get

$$g(S_H \cup \{\emptyset\}) = \sum_{k=1}^{H+1} \beta_k^{(H+1)} \mathbb{E}_{T \in \binom{S_H \cup \{\emptyset\}}{k}} f(T) = \frac{H}{H+1} \beta_1^{(H+1)} + \sum_{k=2}^{H+1} \beta_k^{(H+1)}.$$

Therefore

$$\zeta_H = \frac{H}{H+1} \beta_1^{(H+1)} + \sum_{k=2}^{H+1} \beta_k^{(H+1)}. \quad (10)$$

Equations (9),(10) together imply that for every H ,

$$\zeta_{H+1} = \zeta_H + \frac{\gamma_1^{(H+1)}}{H+1}. \quad (11)$$

Recurrence (11) enables us to prove recurrence (8) for ζ :

$$\zeta_{H+1} = \zeta_H + \frac{\gamma_1^{(H+1)}}{H+1} = \zeta_H + \gamma_1^{(H)} - 1 = \zeta_H + H(\zeta_H - \zeta_{H-1}) - 1 = (H+1)\zeta_H - H\zeta_{H-1} - 1.$$

The first inequality follows from (11), the second from applying (γ -UP) to $\gamma_1^{(H+1)}$ and using the fact that $\gamma_0^{(H+1)} = 1$. The third inequality follows from applying (11) to $\gamma_1^{(H)}$, and the final inequality from simple algebra. The base cases for the recurrence follow directly from (9): $\zeta_0 = 0$ and $\zeta_1 = \beta_1^{(H)} = E - 1$. \square

B Running time of the continuous greedy algorithm

In this section, we analyze the running time of the continuous greedy algorithm [6], as well as a variant which replaces pipage rounding with swap rounding [7]. The algorithm is composed of two parts: the continuous greedy phase and the rounding phase. The second phase is not needed for partition matroids.

We will use T_f to denote the time required to evaluate an oracle call to the submodular function f , and $T_{\mathcal{M}}$ to denote the time required to evaluate an oracle call to the matroid independence oracle. Also, n will be the rank of the matroid \mathcal{M} , u will be the size of the universe, and $\epsilon > 0$ will be a parameter such that the resulting approximation ratio is $1 - 1/e - \epsilon$.

The continuous greedy algorithm uses a continuous relaxation F of the submodular function f . The relaxation $F: [0, 1]^U \rightarrow \mathbb{R}$ is defined as follows:

$$F(y) = \mathbb{E}[f(\hat{y})],$$

where \hat{y} is a random subset of U which contains each $j \in U$ with probability y_j .

Continuous greedy phase The goal of this part is to compute a vector $y \in [0, 1]^U$ such that $F(y) \geq (1 - 1/e - \epsilon)\text{OPT}$. Let $\delta = \epsilon/3n$. We perform $1/\delta$ iterations consisting of the following two steps:

1. For each $j \in U$, estimate $\omega_j = \mathbb{E}[f(\hat{y} \cup \{j\}) - f(\hat{y})]$ by averaging over $O(\delta^{-2} \log u)$ samples.
2. Find a maximum-weight independent set in \mathcal{M} with weights ω_j , and modify y accordingly.

The first step takes time $O(\delta^{-2} u \log u T_f)$. The second step takes time $O(u T_{\mathcal{M}})$. The total running time of this phase is

$$O(\epsilon^{-3} n^3 u \log u T_f + \epsilon^{-1} d u T_{\mathcal{M}}).$$

Rounding phase This stage rounds y into an integral solution z satisfying $f(z) \geq F(y)$. The original paper [6] implements this stage using pipage rounding. Chekuri et al. [7] suggested a simpler and faster implementation, swap rounding. While their algorithm is randomized and only produces a good solution in expectation and with high probability, it is easy to turn it into a deterministic algorithm which always succeeds (this is already mentioned in their paper).

Pipage rounding Pipage rounding consists of up to u^2 iterations of the basic subroutine **HitConstraint**. Each call to **HitConstraint** requires solving the following problem:

$$\delta = \min_{A \in \mathcal{A}} (r_{\mathcal{M}}(A) - y(A)), \quad \mathcal{A} = \{A \subseteq U : i \in A, j \notin A\}.$$

Here i, j are parameters, $r_{\mathcal{M}}(A)$ is the maximal rank of a subset of A , and $y(A) = \sum_{i \in A} y_i$. The time it takes to evaluate $r_{\mathcal{M}}(A) - y(A)$ is $T_r = O(u T_{\mathcal{M}})$.

The paper suggests several methods for finding δ , which amounts to submodular minimization. Among the combinatorial methods, the fastest seems to be the one by Fleischer, Fujishige and Iwata [19], which runs in time $O(u^5 \log n T_r)$. Another method, by Grötschel, Lovász and Schrijver [18], uses the Ellipsoid algorithm with a separation oracle. While this particular approach isn't faster than the combinatorial methods, similar approaches might be, such as using interior-point methods or multiplicative weights. Unfortunately, we have been unable to find a running time analysis in the literature of such methods in this context.

Summarizing, the running time of this phase is

$$O(u^8 \log n T_{\mathcal{M}}).$$

Swap rounding Swap rounding consists of an initialization phase followed by u iterations of the basic subroutine **MergeBases**. The object of the initialization phase is to present y as a convex combination of at most u bases. According to [7], this can be done in time $O(u^6 T_{\mathcal{M}})$. **MergeBases** consists of up to n iterations. Each such iteration takes time $O(n T_{\mathcal{M}} + T_f)$. Thus the running time is dominated by the initialization phase, whose running time is

$$O(u^6 T_{\mathcal{M}}).$$

C Properties of the γ sequence

In this section, we prove the following properties of the γ sequences defined in Section 3:

Lemma 3.1. *The sequences $\gamma^{(m)} = \gamma_0^{(m)}, \dots, \gamma_{m+1}^{(m)}$, satisfy the following properties:*

- (a) $\gamma^{(m)}$ satisfies recurrence (γ -REC) for all $m \geq 1$.
- (b) $\gamma^{(m)}$ is non-decreasing and non-negative for $0 \leq m \leq 2n$.
- (c) $\gamma^{(m-1)}$ and $\gamma^{(m)}$ satisfy both (γ -DOWN) and (γ -UP) for all $m \geq 1$.

Recall that we defined $q = 2 \lfloor \frac{n-1}{2} \rfloor + 1$ and set $\gamma_q^{(2n)} = \gamma_{q+1}^{(2n)} = 1$, then used the recurrence

$$\ell \gamma_{\ell+1}^{(m)} = (2\ell - m + c - 1) \gamma_{\ell}^{(m)} + (m - \ell + 1) \gamma_{\ell-1}^{(m)}, \quad 1 \leq \ell \leq m \quad (\gamma\text{-REC})$$

with $m = 2n$ to define $\gamma_0^{(2n)}, \dots, \gamma_{q-1}^{(2n)}, \gamma_{q+2}^{(2n)}, \dots, \gamma_{2n+1}^{(2n)}$. Next, we normalized this sequence by dividing by $\gamma_0^{(2n)}$ and then used the recurrences

$$\gamma_{\ell}^{(m-1)} = \gamma_0^{(m)} + \frac{c}{m} \sum_{k=1}^{\ell} \gamma_k^{(m)} \quad (\gamma\text{-DOWN})$$

$$\gamma_{\ell+1}^{(m)} = c^{-1} m \left(\gamma_{\ell+1}^{(m-1)} - \gamma_{\ell}^{(m-1)} \right), \quad \gamma_0^{(m)} = \gamma_0^{(m-1)}, \quad \gamma_{m+1}^{(m)} = \gamma_m^{(m-1)} \quad (\gamma\text{-UP})$$

to generate the sequences $\gamma^{(m)}$ for $1 \leq m < 2n$ and $m > 2n$. We now turn to proving various properties of the resulting sequences. In order to prove Lemma 3.1, we first prove the following three lemmas, which will serve as the base case, downward induction step and upward induction step, respectively, of the proof for Lemma C.2.

Lemma C.1. *The sequence $\gamma^{(2n)} = \gamma_0^{(2n)}, \dots, \gamma_{2n+1}^{(2n)}$ is non-decreasing and non-negative.*

Proof. Our proof makes use of the following inequalities, which are immediate from the definition of q :

$$n - 1 \leq q \leq n$$

We consider the sequence $\gamma^{(2n)}$ before it has been normalized by dividing by $\gamma_0^{(2n)}$. In order to prove the lemma, it is sufficient to show that the unnormalized sequence is non-decreasing and $\gamma_0^{(2n)} \geq 0$. Consider the sequence $\gamma^{(2n+1)}$, and note that, for $0 \leq \ell \leq 2n$ we have $\gamma_{\ell+1}^{(2n+1)} \geq 0$ if and only if $\gamma_{\ell+1}^{(2n)} - \gamma_{\ell}^{(2n)} \geq 0$. We prove that $\gamma_{\ell+1}^{(2n+1)} \geq 0$ by induction for ℓ , separately for $\ell \geq q + 1$ and for $\ell \leq q - 1$. By definition, $\gamma_{q+1}^{(2n+1)} = 0$, since $\gamma_q^{(2n)} = \gamma_{q+1}^{(2n)} = 1$.

We begin with the case $\ell \geq q + 1$. Recurrences (γ -REC) and (γ -UP) imply that for ℓ such that $1 \leq \ell \leq 2n$,

$$\begin{aligned} c\ell \gamma_{\ell+1}^{(2n+1)} &= (2n + 1) \left(\ell \gamma_{\ell+1}^{(2n)} - \ell \gamma_{\ell}^{(2n)} \right) \\ &= (2n + 1) \left((2\ell - 2n + c - 1) \gamma_{\ell}^{(2n)} + (2n - \ell + 1) \gamma_{\ell-1}^{(2n)} - \ell \gamma_{\ell}^{(2n)} \right) \\ &= (2n + 1) \left((\ell - 2n + c - 1) (\gamma_{\ell}^{(2n)} - \gamma_{\ell-1}^{(2n)}) + c \gamma_{\ell-1}^{(2n)} \right) \\ &= c(\ell - 2n + c - 1) \gamma_{\ell}^{(2n+1)} + c(2n + 1) \gamma_{\ell-1}^{(2n)}. \end{aligned}$$

Hence, $\gamma_{\ell+1}^{(2n+1)} \geq 0$ if and only if

$$(2n+1)\gamma_{\ell-1}^{(2n)} \geq (2n-\ell-c+1)\gamma_{\ell}^{(2n+1)}. \quad (12)$$

If $\ell = q+1$ then (12) clearly holds since $\gamma_{q+1}^{(2n+1)} = 0$ and $\gamma_q^{(2n)} = 1$. So we assume that $\ell \geq q+2$. We have:

$$c(\ell-1)\gamma_{\ell}^{(2n+1)} = c(\ell-2n+c-2)\gamma_{\ell-1}^{(2n+1)} + c(2n+1)\gamma_{\ell-2}^{(2n)} \leq c(2n+1)\gamma_{\ell-2}^{(2n)} \leq c(2n+1)\gamma_{\ell-1}^{(2n)}.$$

Where we have used $\ell+c \leq 2n+2$ and the induction hypothesis $\gamma_{\ell-1}^{(2n+1)} \geq 0$, which implies $\gamma_{\ell-2}^{(2n)} \leq \gamma_{\ell-1}^{(2n)}$. By the induction hypothesis, we further have $\gamma_{\ell}^{(2n+1)} \geq 0$. Therefore (12) follows from $\ell \geq q+2 \geq n+1$, which implies $2n-\ell-c+1 \leq n-c \leq n \leq \ell-1$.

Consider next the case $\ell \leq q-1$. Recurrence (γ -REC) implies that for ℓ such that $0 \leq \ell \leq 2n-1$,

$$\begin{aligned} c(\ell+1)\gamma_{\ell+2}^{(2n+1)} &= (2n+1) \left((\ell+1)\gamma_{\ell+2}^{(2n)} - (\ell+1)\gamma_{\ell+1}^{(2n)} \right) \\ &= (2n+1) \left((2\ell-2n+c+1)\gamma_{\ell+1}^{(2n)} + (2n-\ell)\gamma_{\ell}^{(2n)} \right) - (\ell+1)\gamma_{\ell+1}^{(2n)} \\ &= (2n+1) \left((\ell-2n)(\gamma_{\ell+1}^{(2n)} - \gamma_{\ell}^{(2n)}) + c\gamma_{\ell+1}^{(2n)} \right) \\ &= c(\ell-2n)\gamma_{\ell+1}^{(2n+1)} + c(2n+1)\gamma_{\ell+1}^{(2n)}, \end{aligned}$$

or, equivalently,

$$(2n-\ell)\gamma_{\ell+1}^{(2n+1)} = (2n+1)\gamma_{\ell+1}^{(2n)} - (\ell+1)\gamma_{\ell+2}^{(2n+1)}.$$

This shows that $\gamma_{\ell+1}^{(2n+1)} \geq 0$ if and only if

$$(2n+1)\gamma_{\ell+1}^{(2n)} \geq (\ell+1)\gamma_{\ell+2}^{(2n+1)}. \quad (13)$$

If $\ell = q-1$ then (13) clearly holds since $\gamma_{q+1}^{(2n+1)} = 0$ and $\gamma_q^{(2n)} = 1$. So we assume that $\ell \leq q-2$. We have:

$$(2n-\ell-1)\gamma_{\ell+2}^{(2n+1)} = (2n+1)\gamma_{\ell+2}^{(2n)} - (\ell+2)\gamma_{\ell+3}^{(2n+1)} \leq (2n+1)\gamma_{\ell+2}^{(2n)},$$

since $\gamma_{\ell+3}^{(2n+1)} \geq 0$ by the induction hypothesis. Subtracting $c\gamma_{\ell+2}^{(2n+1)} = (2n+1)(\gamma_{\ell+2}^{(2n)} - \gamma_{\ell+1}^{(2n)})$, which follows from (γ -UP), we obtain

$$(2n-\ell-1-c)\gamma_{\ell+2}^{(2n+1)} \leq (2n+1)\gamma_{\ell+1}^{(2n)}.$$

From the induction hypothesis, $\gamma_{\ell+2}^{(2n+1)} \geq 0$. Therefore, (13) follows from $\ell \leq q-2 \leq n-2$, which implies that $\ell+1 \leq n-1 \leq n+1-c \leq 2n-\ell-1-c$.

We have shown that $\gamma_{\ell}^{(2n+1)} \geq 0$ for all $1 \leq \ell \leq 2n+1$, and so (12) and (13) imply that $\gamma_{\ell}^{(2n)} \geq 0$ for $1 \leq \ell \leq 2n+1$. We now show that $\gamma_0^{(2n)} \geq 0$ as well. From the recurrence (γ -REC) we have:

$$\gamma_2^{(2n)} = (1+c-2n)\gamma_1^{(2n)} + 2n\gamma_0^{(2n)}.$$

Since $\gamma_2^{(2n)} \geq 0$ and $\gamma_1^{(2n)} \geq 0$,

$$\gamma_0^{(2n)} = \frac{(2n-1-c)\gamma_1^{(2n)} + \gamma_2^{(2n)}}{2n} \geq 0. \quad \square$$

Lemma C.2. Suppose $\gamma^{(m)} = \gamma_0^{(m)}, \dots, \gamma_{m+1}^{(m)}$ is a non-decreasing, non-negative sequence satisfying $(\gamma\text{-REC})$, and define $\gamma^{(m-1)} = \gamma_0^{(m-1)}, \dots, \gamma_m^{(m-1)}$ using $(\gamma\text{-DOWN})$. Then,

- (a) If $m \geq 2$, the sequence $\gamma^{(m-1)}$ satisfies recurrence $(\gamma\text{-REC})$.
- (b) The sequences $\gamma^{(m)}$ and $\gamma^{(m-1)}$ satisfy the recurrence $(\gamma\text{-UP})$.
- (c) The sequence $\gamma^{(m-1)}$ is non-decreasing and non-negative.

Proof. For part (a), let k satisfy $1 \leq k \leq m-1$. We must show

$$k\gamma_{k+1}^{(m-1)} = (2k - m + c)\gamma_k^{(m-1)} + (m - k)\gamma_{k-1}^{(m-1)}.$$

After applying $(\gamma\text{-DOWN})$ and multiplying by m/c , this is equivalent to

$$k\gamma_{k+1}^{(m)} = (k - m + c)\gamma_k^{(m)} + c \sum_{k=1}^{k-1} \gamma_k^{(m)} + m\gamma_0^{(m)}.$$

Summing the recurrence $(\gamma\text{-REC})$ for $\ell = 1, \dots, k$ yields exactly this equation.

For part (b), we begin by considering the base cases for recurrence $(\gamma\text{-UP})$. It follows directly from $(\gamma\text{-DOWN})$ that $\gamma_0^{(m-1)} = \gamma_0^{(m)}$. Summing recurrence $(\gamma\text{-REC})$ for $\gamma^{(m)}$ for $\ell = 1, \dots, m$, we get

$$m\gamma_{m+1}^{(m)} = m\gamma_0^{(m)} + c \sum_{k=1}^m \gamma_k^{(m)} = m\gamma_m^{(m-1)}.$$

Next, we must show that

$$\gamma_{\ell+1}^{(m)} = c^{-1}m \left(\gamma_{\ell+1}^{(m-1)} - \gamma_{\ell}^{(m-1)} \right)$$

for $1 \leq \ell \leq m-1$. After applying $(\gamma\text{-DOWN})$ and simplifying, this is equivalent to showing

$$\gamma_{\ell+1}^{(m)} = c^{-1}m \frac{c}{m} \gamma_{\ell+1}^{(m)}.$$

For part (c), we note that $\gamma^{(m)}$ is non-negative, and from part (b) we have $\gamma_{\ell+1}^{(m)} \geq 0$ if and only if $\gamma_{\ell+1}^{(m-1)} - \gamma_{\ell}^{(m-1)} \geq 0$. Thus, the sequence $\gamma^{(m-1)}$ is non-decreasing. Furthermore, from part (a), $\gamma_0^{(m-1)} = \gamma_0^{(m)}$, which is non-negative. Thus, the entire sequence $\gamma^{(m-1)}$ is non-negative. \square

Lemma C.3. Suppose $\gamma^{(m-1)} = \gamma_0^{(m-1)}, \dots, \gamma_m^{(m-1)}$ is a sequence satisfying recurrence $(\gamma\text{-REC})$. Define $\gamma^{(m)} = \gamma_0^{(m)}, \dots, \gamma_{m+1}^{(m)}$ using $(\gamma\text{-UP})$. Then,

- (a) The sequence $\gamma^{(m)}$ satisfies recurrence $(\gamma\text{-REC})$.
- (b) The sequences $\gamma^{(m)}$ and $\gamma^{(m-1)}$ satisfy the recurrence $(\gamma\text{-DOWN})$.

Proof. We start by proving part (a). When $\ell = 1$, we have

$$\begin{aligned} \gamma_2^{(m)} &= c^{-1}m(\gamma_2^{(m-1)} - \gamma_1^{(m-1)}) && \text{by } (\gamma\text{-UP}) \\ &= c^{-1}m((2 - m + c)\gamma_1^{(m-1)} + (m - 1)\gamma_0^{(m-1)} - \gamma_1^{(m-1)}) && \text{by } (\gamma\text{-REC}) \\ &= c^{-1}m((1 - m + c)\gamma_1^{(m-1)} - (1 - m + c)\gamma_0^{(m-1)} + c\gamma_0^{(m-1)}) && \text{by algebra} \\ &= (1 - m + c)\gamma_1^{(m)} + m\gamma_0^{(m-1)} && \text{by } (\gamma\text{-UP}) \\ &= (1 - m + c)\gamma_1^{(m)} + m\gamma_0^{(m)} && \text{by } (\gamma\text{-UP}). \end{aligned}$$

When $\ell = m$, we have

$$\begin{aligned}
m\gamma_{m+1}^{(m)} &= m\gamma_m^{(m-1)} && \text{by } (\gamma\text{-UP}) \\
&= c^{-1}m((m-1+c)\gamma_m^{(m-1)} - (m-1)\gamma_m^{(m-1)}) && \text{by algebra} \\
&= c^{-1}m((m-1+c)\gamma_m^{(m-1)} - (m-2+c)\gamma_{m-1}^{(m-1)} - \gamma_{m-2}^{(m-1)}) && \text{by } (\gamma\text{-REC}) \\
&= c^{-1}m((m-1+c)(\gamma_m^{(m-1)} - \gamma_{m-1}^{(m-1)}) + \gamma_{m-1}^{(m-1)} - \gamma_{m-2}^{(m-1)}) && \text{by algebra} \\
&= (m-1+c)\gamma_m^{(m)} + \gamma_{m-1}^{(m)} && \text{by } (\gamma\text{-UP}) \text{ twice.}
\end{aligned}$$

When $2 \leq \ell \leq m-1$, we have

$$\begin{aligned}
\ell\gamma_{\ell+1}^{(m)} &= c^{-1}m(\ell\gamma_{\ell+1}^{(m-1)} - \ell\gamma_{\ell}^{(m-1)}) && \text{by } (\gamma\text{-UP}) \\
&= c^{-1}m((2\ell-m+c)\gamma_{\ell}^{(m-1)} + (m-\ell)\gamma_{\ell-1}^{(m-1)} - \ell\gamma_{\ell}^{(m-1)}) && \text{by } (\gamma\text{-REC}) \\
&= c^{-1}m((2\ell-m+c-1)\gamma_{\ell}^{(m-1)} + (m-\ell)\gamma_{\ell-1}^{(m-1)} - (\ell-1)\gamma_{\ell}^{(m-1)}) && \text{by algebra} \\
&= c^{-1}m((2\ell-m+c-1)\gamma_{\ell}^{(m-1)} + (m-\ell)\gamma_{\ell-1}^{(m-1)} && \text{by } (\gamma\text{-REC}) \\
&\quad - (2\ell-m+c-2)\gamma_{\ell-1}^{(m-1)} - (m-\ell+1)\gamma_{\ell-2}^{(m-1)}) \\
&= c^{-1}m((2\ell-m+c-1)\gamma_{\ell}^{(m-1)} + (m-\ell+1)\gamma_{\ell-1}^{(m-1)} && \text{by algebra} \\
&\quad - (2\ell-m+c-1)\gamma_{\ell-1}^{(m-1)} - (m-\ell+1)\gamma_{\ell-2}^{(m-1)}) \\
&= (2\ell-m+c-1)\gamma_{\ell}^{(m)} + (m-\ell+1)\gamma_{\ell-1}^{(m)}. && \text{by } (\gamma\text{-UP}) \text{ twice}
\end{aligned}$$

For part (b) we must show that for $0 \leq \ell \leq m$,

$$\gamma_{\ell}^{(m-1)} = \gamma_0^{(m)} + \frac{c}{m} \sum_{k=1}^{\ell} \gamma_k^{(m)}$$

Applying $(\gamma\text{-UP})$, this is equivalent to showing that:

$$\gamma_{\ell}^{(m-1)} = \gamma_0^{(m-1)} + \sum_{k=1}^{\ell} \left(\gamma_k^{(m-1)} - \gamma_{k-1}^{(m-1)} \right) = \gamma_{\ell}^{(m-1)}. \quad \square$$

We are now ready to prove the properties of the γ sequences stated in Lemma 3.1:

Proof of Lemma 3.1. By definition, $\gamma^{(2n)}$ satisfies recurrence $(\gamma\text{-REC})$. Inductively applying Lemma C.2 (a) for $m = 2n, 2n-1, \dots, 1$ and Lemma C.3 (a) for $m = 2n+1, 2n+2, \dots$ then completes the proof of part (a).

By Lemma C.1, $\gamma^{(2n)}$ is non-decreasing and non-negative. Inductively applying Lemma C.2 (c) for $m = 2n, 2n-1, \dots, 1$ then proves part (b).

For part (c), we note that for all sequences $\gamma^{(m-1)}, \gamma^{(m)}$ with $m \leq 2n$, $(\gamma\text{-DOWN})$ holds by construction. Inductively applying Lemma C.2 (b) then shows that $(\gamma\text{-UP})$ holds. Similarly, if $m > 2n$, then $(\gamma\text{-UP})$ holds by construction and inductively applying Lemma C.3 (b) shows that $(\gamma\text{-DOWN})$ holds. \square

Lemma 3.2. For all $m \geq 0$, the terms of $\gamma^{(m)}$ are given by the following formula:

$$\gamma_{\ell+1}^{(m)} = (-1)^\ell \sum_{k=0}^{m-1} \frac{m!}{k!} c^{k-m} \left[(-1)^k \binom{m-1-k}{\ell-k} E - \binom{m-1-k}{\ell} \right] \quad (\gamma\text{-CLOSED})$$

where $E = \gamma_{m+1}^{(m)}$ and $0 \leq \ell \leq m-1$.

Proof. We proceed by induction on m . When $m = 0$, the statement is trivial, as we have only $\gamma_0^{(0)} = 1$ and $\gamma_1^{(0)} = E$. If $m \geq 1$, then by Lemma 3.1, the sequence $\gamma^{(m-1)}$ is related to $\gamma^{(m)}$ by (γ -UP). Applying the induction hypothesis to $\gamma^{(m-1)}$, we find that $\gamma_{\ell+1}^{(m)}$ is equal to

$$\begin{aligned} & c^{-1}m \left(\gamma_{\ell+1}^{(m-1)} - \gamma_\ell^{(m-1)} \right) \\ &= c^{-1}m(-1)^\ell \sum_{k=0}^{m-2} \frac{(m-1)!}{k!} c^{k-m+1} \left[(-1)^k \binom{m-2-k}{\ell-k} E - \binom{m-2-k}{\ell} \right] \\ &\quad - c^{-1}m(-1)^{\ell-1} \sum_{k=0}^{m-2} \frac{(m-1)!}{k!} c^{k-m+1} \left[(-1)^k \binom{m-2-k}{\ell-1-k} E - \binom{m-2-k}{\ell-1} \right] \\ &= (-1)^\ell \sum_{k=0}^{m-2} \frac{m!}{k!} c^{k-m} \left[(-1)^k \left[\binom{m-2-k}{\ell-k} + \binom{m-2-k}{\ell-1-k} \right] E - \left[\binom{m-2-k}{\ell} + \binom{m-2-k}{\ell-1} \right] \right] \\ &= (-1)^\ell \sum_{k=0}^{m-2} \frac{m!}{k!} c^{k-m} \left[(-1)^k \binom{m-1-k}{\ell-k} E - \binom{m-1-k}{\ell} \right]. \end{aligned}$$

If $1 \leq \ell \leq m-2$, then we have both $\binom{m-1-(m-1)}{\ell-(m-1)} = 0$ and $\binom{m-1-(m-1)}{\ell} = 0$, and so:

$$\begin{aligned} \gamma_{\ell+1}^{(m)} &= (-1)^\ell \sum_{k=0}^{m-2} \frac{m!}{k!} c^{k-m} \left[(-1)^k \binom{m-1-k}{\ell-k} E - \binom{m-1-k}{\ell} \right] \\ &= (-1)^\ell \sum_{k=0}^{m-1} \frac{m!}{k!} c^{k-m} \left[(-1)^k \binom{m-1-k}{\ell-k} E - \binom{m-1-k}{\ell} \right]. \end{aligned}$$

It remains to show that the formula holds for $\ell = 0$ and for $\ell = m-1$. In the case of $\ell = 0$, (γ -CLOSED) gives

$$\gamma_1^{(m)} = (-1)^0 \sum_{k=0}^{m-1} \frac{m!}{k!} c^{k-m} \left[(-1)^k \binom{m-1-k}{-k} E - \binom{m-1-k}{0} \right] = m!c^{-m}E - \sum_{k=0}^{m-1} \frac{m!}{k!} c^{k-m},$$

while using the recurrence (γ -UP) and the induction hypothesis, we obtain

$$\begin{aligned}
\gamma_1^{(m)} &= c^{-1}m \left(\gamma_1^{(m-1)} - \gamma_0^{(m-1)} \right) = c^{-1}m \left(\gamma_1^{(m-1)} - 1 \right) \\
&= c^{-1}m(-1)^0 \sum_{k=0}^{m-2} \frac{(m-1)!}{k!} c^{k-m+1} \left[(-1)^k \binom{m-2-k}{-k} E - \binom{m-2-k}{0} \right] - c^{-1}m \\
&= m!c^{-m}E - \sum_{k=0}^{m-2} \frac{m!}{k!} c^{k-m} - c^{-1}m \\
&= m!c^{-m}E - \sum_{k=0}^{m-1} \frac{m!}{k!} c^{k-m} .
\end{aligned}$$

In the case $\ell = m-1$, (γ -CLOSED) gives

$$\begin{aligned}
\gamma_m^{(m)} &= (-1)^{m-1} \sum_{k=0}^{m-1} \frac{m!}{k!} c^{k-m} \left[(-1)^k \binom{m-1-k}{m-1-k} E - \binom{m-1-k}{m-1} \right] \\
&= (-1)^{m-1} \left[\left(\sum_{k=0}^{m-1} \frac{m!}{k!} c^{k-m} (-1)^k E \right) - m!c^{-m} \right],
\end{aligned}$$

while using the recurrence (γ -UP) and the induction hypothesis, we obtain

$$\begin{aligned}
\gamma_m^{(m)} &= c^{-1}m \left(\gamma_m^{(m-1)} - \gamma_{m-1}^{(m-1)} \right) = c^{-1}m \left(E - \gamma_{m-1}^{(m-1)} \right) \\
&= c^{-1}mE - c^{-1}m(-1)^{m-2} \sum_{k=0}^{m-2} \frac{(m-1)!}{k!} c^{k-m+1} \left[(-1)^k \binom{m-2-k}{m-2-k} E - \binom{m-2-k}{m-2} \right] \\
&= c^{-1}mE + (-1)^{m-1} \left[\left(\sum_{k=0}^{m-2} \frac{m!}{k!} c^{k-m} (-1)^k E \right) - m!c^{-m} \right] \\
&= (-1)^{m-1} \left[\left(\sum_{k=0}^{m-1} \frac{m!}{k!} c^{k-m} (-1)^k E \right) - m! \right]. \quad \square
\end{aligned}$$

The next identity will be useful in proving some of the remaining results in the Appendix.

Lemma C.4. *For all $m \geq 1$:*

$$\sum_{k=1}^m \gamma_k^{(m)} = m\gamma_1^{(1)}$$

Proof. We have

$$\sum_{k=1}^m \gamma_k^{(m)} = \sum_{k=1}^m c^{-1}m \left(\gamma_k^{(m-1)} - \gamma_{k-1}^{(m-1)} \right) = c^{-1}m \left(\gamma_m^{(m-1)} - \gamma_0^{(m-1)} \right) = c^{-1}m(\gamma_2^{(1)} - \gamma_0^{(1)}),$$

where the first and third equalities follow from Lemma 3.1 (c). Furthermore, recurrence (γ -REC) for $m=1$ states that $\gamma_2^{(1)} - \gamma_0^{(1)} = c\gamma_1^{(1)}$. \square

Finally, we briefly verify the equivalence between the explicit formula (γ -CLOSED) and the expression (7) from Section 4.2.

Lemma C.5.

$$\gamma_{\ell+1}^{(m)} = (-1)^\ell m! c^{-m} \binom{m-1}{\ell} [Q_{m-1-\ell, \ell}(c)E - P_{m-1-\ell, \ell}(c)]. \quad (7)$$

Proof. From (γ -CLOSED) we have

$$\begin{aligned} \gamma_{\ell+1}^{(m)} &= (-1)^\ell \sum_{k=0}^{m-1} \frac{m!}{k!} c^{k-m} \left[(-1)^k \binom{m-1-k}{\ell-k} E - \binom{m-1-k}{\ell} \right] \\ &= (-1)^\ell m! c^{-m} \sum_{k=0}^{m-1} \frac{c^k}{k!} \left[(-1)^k \binom{m-1-k}{\ell-k} E - \binom{m-1-k}{\ell} \right] \\ &= (-1)^\ell m! c^{-m} \left[\sum_{k=0}^{\ell} \frac{c^k}{k!} (-1)^k \binom{m-1-k}{\ell-k} E - \sum_{k=0}^{m-1-\ell} \frac{c^k}{k!} \binom{m-1-k}{\ell} \right] \\ &= (-1)^\ell m! c^{-m} \left[\sum_{k=0}^{\ell} \frac{(-c)^k (m-1-k)!}{k! (m-1-\ell)! (\ell-k)!} E - \sum_{k=0}^{m-1-\ell} \frac{c^k (m-1-k)!}{k! (m-1-\ell-k)! \ell!} \right] \\ &= (-1)^\ell m! c^{-m} \frac{(m-1)!}{(m-1-\ell)! \ell!} \left[\sum_{k=0}^{\ell} \frac{(-c)^k (m-1-k)! \ell!}{k! (m-1)! (\ell-k)!} E - \sum_{k=0}^{m-1-\ell} c^k \frac{(m-1-k)! (m-1-\ell)!}{k! (m-1)! (m-1-\ell-k)!} \right] \\ &= (-1)^\ell m! c^{-m} \binom{m-1}{\ell} [Q_{m-1-\ell, \ell}(c)E - P_{m-1-\ell, \ell}(c)]. \quad \square \end{aligned}$$

D Properties of g

Here we prove that g has the properties claimed in Section 3. We begin by proving a small identity.

Lemma D.1. *For all $1 \leq \ell \leq m$:*

$$(m+1)\beta_\ell^{(m)} = (m+1-\ell)\beta_\ell^{(m+1)} + (\ell+1)\beta_{\ell+1}^{(m+1)}$$

Proof. The sequences $\gamma^{(m)}$ and $\gamma^{(m+1)}$ satisfy (γ -REC) and are related by (γ -UP), as shown in Lemma 3.1. Multiplying the identity by ℓ and using the definition $\beta_k^{(m)} = k\gamma_k^{(m)}$, we obtain

$$(m+1)\gamma_\ell^{(m)} = (m+1-\ell)\gamma_\ell^{(m+1)} + \ell\gamma_{\ell+1}^{(m+1)}.$$

Applying (γ -UP) to each term on the right the desired identity is equivalent to

$$\begin{aligned} (m+1)\gamma_\ell^{(m)} &= c^{-1}(m+1) \left[(m+1-\ell)(\gamma_\ell^{(m)} - \gamma_{\ell-1}^{(m)}) + \ell(\gamma_{\ell+1}^{(m)} - \gamma_\ell^{(m)}) \right] \\ &= c^{-1}(m+1) \left[\ell\gamma_{\ell+1}^{(m)} + (m-2\ell+1)\gamma_\ell^{(m)} - (m-\ell+1)\gamma_{\ell-1}^{(m)} \right] \\ &= (m+1)\gamma_\ell^{(m)}, \end{aligned}$$

where we have used the recurrence (γ -REC) in the final line. \square

D.1 Monotonicity

Lemma D.2. *For any set S of size $m < 2n$ and $x \notin S$, $g(S) \leq g(S \cup \{x\})$.
Moreover, if $f(T) = f(T \cup \{x\})$ for each $T \subseteq S$ then $g(S) = g(S \cup \{x\})$.*

Proof. Define for $1 \leq k \leq m$,

$$p_k = \left(1 - \frac{k}{m+1}\right) \frac{\beta_k^{(m+1)}}{\beta_k^{(m)}},$$

$$q_k = \frac{k+1}{m+1} \frac{\beta_{k+1}^{(m+1)}}{\beta_k^{(m)}}.$$

Lemma D.1 implies that $p_k + q_k = 1$. Since $\gamma^{(m)}$ and $\gamma^{(m+1)}$ are non-negative from Lemma 3.1 part (c), $\beta^{(m)}$ and $\beta^{(m+1)}$ are non-negative and thus $0 \leq p_k, q_k \leq 1$. The following identities will be useful:

$$p_k \frac{\beta_k^{(m)}}{\binom{m}{k}} = \beta_k^{(m+1)} \frac{m+1-k}{m+1} \frac{1}{\binom{m}{k}} = \frac{\beta_k^{(m+1)}}{\binom{m+1}{k}},$$

$$q_{k-1} \frac{\beta_{k-1}^{(m)}}{\binom{m}{k-1}} = \beta_k^{(m+1)} \frac{k}{m+1} \frac{1}{\binom{m}{k-1}} = \frac{\beta_k^{(m+1)}}{\binom{m+1}{k}}.$$

We have

$$\begin{aligned} g(S) &= \sum_{k=1}^m \beta_k^{(m)} \mathbb{E}_{\substack{T \subseteq S \\ |T|=k}} f(T) \\ &= \sum_{k=1}^m \frac{\beta_k^{(m)}}{\binom{m}{k}} \sum_{\substack{T \subseteq S \\ |T|=k}} f(T) \\ &\leq \left[\sum_{k=1}^m \frac{\beta_k^{(m)}}{\binom{m}{k}} \sum_{\substack{T \subseteq S \\ |T|=k}} (p_k f(T) + q_k f(T \cup \{x\})) \right] + \beta_1^{(m+1)} f(\{x\}) \\ &= \left[\sum_{k=1}^m p_k \frac{\beta_k^{(m)}}{\binom{m}{k}} \sum_{\substack{T \subseteq S \\ |T|=k}} f(T) \right] + \left[\sum_{k=2}^{m+1} q_{k-1} \frac{\beta_{k-1}^{(m)}}{\binom{m}{k-1}} \sum_{\substack{T \subseteq S \\ |T|=k-1}} f(T \cup \{x\}) \right] + \beta_1^{(m+1)} f(\{x\}) \\ &= \sum_{k=1}^{m+1} \beta_k^{(m+1)} \mathbb{E}_{\substack{T \subseteq S \cup \{x\} \\ |T|=k}} f(T) = g(S \cup \{x\}). \end{aligned}$$

If $f(T \cup \{x\}) = f(T)$ for all $T \subseteq S$ then all the inequalities become tight. □

D.2 Submodularity

We first prove another small identity:

$$\frac{2\beta_k^{(m)}}{\binom{m}{k}} + \frac{\beta_{k+2}^{(m+1)}}{\binom{m+1}{k+2}} = \frac{\beta_k^{(m-1)}}{\binom{m-1}{k}} + \frac{\beta_k^{(m+1)}}{\binom{m+1}{k}}. \quad (14)$$

Proof. Multiplying the desired identity by $(m-k)(m+1)\binom{m}{k}$, we obtain the equivalent statement

$$2(m-k)(m+1)\beta_k^{(m)} + (k+1)(k+2)\beta_{k+2}^{(m+1)} = m(m+1)\beta_k^{(m-1)} + (m-k)(m-k+1)\beta_k^{(m+1)}.$$

Applying Lemma (D.1) to the left yields

$$2(m-k)(m+1-k)\beta_k^{(m+1)} + 2(m-k)(k+1)\beta_{k+1}^{(m+1)} + (k+1)(k+2)\beta_{k+2}^{(m+1)}.$$

Applying Lemma (D.1) twice to the right yields

$$(m-k)(m+1-k)\beta_k^{(m+1)} + (m-k)(k+1)\beta_{k+1}^{(m+1)} + (k+1)(m-k)\beta_{k+1}^{(m+1)} \\ + (k+1)(k+2)\beta_{k+2}^{(m+1)} + (m-k)(m+1-k)\beta_k^{(m+1)},$$

which is the same as the expression for the left. \square

Lemma D.3. *For any set S of size $m-1 < 2n-1$ and $x \neq y \notin S$, we have*

$$g(S \cup \{x\}) + g(S \cup \{y\}) \geq g(S \cup \{x, y\}) + g(S).$$

Proof. Define p_k, q_k for k satisfying $1 \leq k \leq 2n-1$ as in the proof of Lemma D.2. As in the lemma, we have $p_k + q_k = 1$ and $0 \leq p_k, q_k \leq 1$. Then, the identity for q_k from Lemma D.2 and identity (14) show that:

$$2\frac{\beta_k^{(m)}}{\binom{m}{k}} + q_{k+1}\frac{\beta_{k+1}^{(m)}}{\binom{m}{k+1}} = 2\frac{\beta_k^{(m)}}{\binom{m}{k}} + \frac{\beta_{k+2}^{(m)}}{\binom{m+1}{k+2}} = \frac{\beta_k^{(m-1)}}{\binom{m-1}{k}} + \frac{\beta_k^{(m+1)}}{\binom{m+1}{k}}.$$

We have

$$\begin{aligned} g(S \cup \{x\}) + g(S \cup \{y\}) &= \sum_{k=1}^{m-1} \frac{\beta_k^{(m)}}{\binom{m}{k}} \sum_{\substack{T \subseteq S \\ |T|=k}} 2f(T) + \sum_{k=1}^m \frac{\beta_k^{(m)}}{\binom{m}{k}} \sum_{\substack{T \subseteq S \\ |T|=k-1}} (f(T \cup \{x\}) + f(T \cup \{y\})) \\ &\geq \sum_{k=1}^{m-1} \frac{\beta_k^{(m)}}{\binom{m}{k}} \sum_{\substack{T \subseteq S \\ |T|=k}} 2f(T) + \sum_{k=1}^m \frac{\beta_k^{(m)}}{\binom{m}{k}} \sum_{\substack{T \subseteq S \\ |T|=k-1}} [p_k(f(T \cup \{x\}) + f(T \cup \{y\})) + q_k(f(T) + f(T \cup \{x, y\}))] \\ &= \sum_{k=1}^{m-1} \left(2\frac{\beta_k^{(m)}}{\binom{m}{k}} + q_{k+1}\frac{\beta_{k+1}^{(m)}}{\binom{m}{k+1}} \right) \sum_{\substack{T \subseteq S \\ |T|=k}} f(T) + q_1 \frac{\beta_1^{(m)}}{\binom{m}{1}} f(\emptyset) \\ &\quad + \sum_{k=1}^m p_k \frac{\beta_k^{(m)}}{\binom{m}{k}} \sum_{\substack{T \subseteq S \\ |T|=k-1}} (f(T \cup \{x\}) + f(T \cup \{y\})) + \sum_{k=2}^{m+1} q_{k-1} \frac{\beta_{k-1}^{(m)}}{\binom{m}{k-1}} \sum_{\substack{T \subseteq S \\ |T|=k-2}} f(T \cup \{x, y\}) \end{aligned}$$

$$\begin{aligned}
&= \sum_{k=1}^{m-1} \left(2 \frac{\beta_k^{(m)}}{\binom{m}{k}} + q_{k+1} \frac{\beta_{k+1}^{(m)}}{\binom{m}{k+1}} \right) \sum_{\substack{T \subseteq S \\ |T|=k}} f(T) + \sum_{k=1}^m p_k \frac{\beta_k^{(m)}}{\binom{m}{k}} \sum_{\substack{T \subseteq S \\ |T|=k-1}} (f(T \cup \{x\}) + f(T \cup \{y\})) \\
&\quad + \sum_{k=2}^{m+1} q_{k-1} \frac{\beta_{k-1}^{(m)}}{\binom{m}{k-1}} \sum_{\substack{T \subseteq S \\ |T|=k-2}} f(T \cup \{x, y\}) \\
&= \sum_{k=1}^{m-1} \left(\frac{\beta_k^{(m-1)}}{\binom{m-1}{k}} + \frac{\beta_k^{(m+1)}}{\binom{m+1}{k}} \right) \sum_{\substack{T \subseteq S \\ |T|=k}} f(T) + \sum_{k=1}^m \frac{\beta_k^{(m+1)}}{\binom{m+1}{k}} \sum_{\substack{T \subseteq S \\ |T|=k-1}} (f(T \cup \{x\}) + f(T \cup \{y\})) \\
&\quad + \sum_{k=2}^{m+1} \frac{\beta_k^{(m+1)}}{\binom{m+1}{k}} \sum_{\substack{T \subseteq S \\ |T|=k-2}} f(T \cup \{x, y\}) \\
&= g(S) + g(S \cup \{x, y\}). \quad \square
\end{aligned}$$

D.3 Curvature

Lemma D.4. For any set S of size $m < 2n$ and $x \notin S$, if $f(T \cup \{x\}) \geq f(T) + (1-c)f(\{x\})$ for all $T \subseteq S$,

$$g(S \cup \{x\}) \geq g(S) + (1-c')g(\{x\}), \quad c' = c \left(1 - \frac{\beta_1^{(n+1)}}{(n+1)\beta_1^{(1)}} \right).$$

Proof. Lemma D.1 implies that for k satisfying $1 \leq k \leq m$,

$$\frac{\beta_k^{(m+1)}}{\binom{m+1}{k}} + \frac{\beta_{k+1}^{(m+1)}}{\binom{m+1}{k+1}} = \frac{\beta_k^{(m)}}{\binom{m}{k}}.$$

Using this and Lemma C.4, we have

$$\begin{aligned}
g(S \cup x) &= \sum_{k=1}^{m+1} \frac{\beta_k^{(m+1)}}{\binom{m+1}{k}} \left[\sum_{\substack{T \subseteq S \\ |T|=k}} f(T) + \sum_{\substack{T \subseteq S \\ |T|=k-1}} f(T \cup \{x\}) \right] \\
&\geq \left[\sum_{k=1}^{m+1} \frac{\beta_k^{(m+1)}}{\binom{m+1}{k}} \left[\sum_{\substack{T \subseteq S \\ |T|=k}} f(T) + \sum_{\substack{T \subseteq S \\ |T|=k-1}} (f(T) + (1-c)f(\{x\})) \right] \right] + \frac{c\beta_1^{(m+1)}}{m+1} f(\{x\}) \\
&= \left[\sum_{k=1}^m \left(\frac{\beta_k^{(m+1)}}{\binom{m+1}{k}} + \frac{\beta_{k+1}^{(m+1)}}{\binom{m+1}{k+1}} \right) \sum_{\substack{T \subseteq S \\ |T|=k}} f(T) \right] + (1-c) \left[\sum_{k=1}^{m+1} \frac{\binom{m}{k-1}}{\binom{m+1}{k}} \beta_k^{(m+1)} f(\{x\}) \right] + \frac{c\beta_1^{(m+1)}}{m+1} f(\{x\})
\end{aligned}$$

$$\begin{aligned}
&= \left[\sum_{k=1}^m \frac{\beta_k^{(m)}}{\binom{m}{k}} \sum_{\substack{T \subseteq S \\ |T|=k}} f(T) \right] + (1-c) \left[\sum_{k=1}^{m+1} \frac{k\beta_k^{(m+1)}}{m+1} f(\{x\}) \right] + \frac{c\beta_1^{(m+1)}}{m+1} f(\{x\}) \\
&= g(S) + (1-c)\beta_1^{(1)} f(\{x\}) + \frac{c\beta_1^{(m+1)}}{m+1} f(\{x\}) \\
&= g(S) + (1-c)g(\{x\}) + \frac{c\beta_1^{(m+1)}}{(m+1)\beta_1^{(1)}} g(\{x\}). \quad \square
\end{aligned}$$

Note that $\beta_1^{(m+1)} > 0$, and so $c' < c$. We can think of the construction of g from f as an operator G on the space of functions. The lemma shows that the curvature of $G^{(t)}(f)$ (that is, G applied t times in a row to f) tends to zero with t , and so in the limit we get a linear function which necessarily depends only on the value of f on singletons. On the other hand, when m is large, $\beta_1^{(m+1)} \approx 1$, and so $c' \approx c$.

E Sampling g

Here we prove Lemma 3.3. Our analysis makes use of the following lemma, which is interesting in its own right.

Lemma E.1. *Let f be a non-negative submodular function, and let S be a set of size m . For k in the range $1 \leq k \leq m$, define $F(k) = \mathbb{E} f(X_k)$, where X_k is a uniformly random subset of S of size k . Then*

$$F(k) \geq \frac{k}{m} F(m).$$

Proof. The proof is by induction on k . For $k = 1$ we have

$$mF(1) = \sum_{x \in S} f(\{x\}) \geq f(S),$$

from submodularity of f . Now, suppose that the claim is true for $k - 1$. Let $S = \{s_1, \dots, s_m\}$. From submodularity we have

$$\sum_{t=k}^m f(\{s_1, \dots, s_{k-1}, s_t\}) \geq (m-k)f(\{s_1, \dots, s_{k-1}\}) + f(S).$$

Averaging over all permutations of indices and then applying the induction hypothesis, we obtain

$$(m-k+1)F(k) \geq (m-k)F(k-1) + f(S) \geq (m-k)\frac{k-1}{m}f(S) + f(S) = (m-k+1)\frac{k}{m}f(S). \quad \square$$

Recall that we made use of the following random process for estimating $g(S)$. Suppose that $|S| = m \leq 2n$, and define $\alpha_m = \sum_{k=1}^m \beta_k^{(m)}$. We define the random set X using the following two step experiment. First, let L be a random variable taking value k with probability $\beta_k^{(m)}/\alpha_m$, then choose X as a uniformly random subset of S of size L . Then, from the linearity of expectations we have $\alpha_m \mathbb{E} f(X) = g(S)$.

We need the following upper bound on α_m .

Lemma E.2. *If $2 \leq m \leq 2n$ then*

$$\alpha_m = \sum_{k=1}^m \beta_k^{(m)} \leq 8 \log m.$$

Proof. From Lemma 3.1 part (b), we must have $\gamma^{(2)}$ non-decreasing. Using the explicit formula (γ -CLOSED) for $\gamma^{(2)}$ we find $\gamma_1^{(2)} = 2c^{-2}(cE - 1 - c)$ and $\gamma_2^{(2)} = 2c^{-2}(1 - (1 - c)E)$. Algebra gives $E \leq (2 + c)/(2 - c) \leq 3$. Additionally, because $\gamma^{(m)}$ is non-decreasing, we have $\gamma_k^{(m)} \leq \gamma_{m+1}^{(m)} = E$ for all $k \leq m + 1$. Therefore, using $m \geq 2$,

$$\alpha_m = \sum_{k=1}^m \beta_k^{(m)} = \sum_{k=1}^m \frac{\gamma_k^{(m)}}{k} \leq E \sum_{k=1}^m \frac{1}{k} \leq 3(\log m + 1) \leq 8 \log m. \quad (15) \quad \square$$

We now prove E.1, showing that we can estimate g reasonably well by sampling from this random process.

Lemma 3.3. *Let S be a set of size m . Let N be a positive integer, and X_1, \dots, X_N be N i.i.d. random samples drawn from the distribution for X . Define $\tilde{g} = \frac{1}{N} \sum_{i=1}^N \alpha_m f(X_i)$. For every $\epsilon > 0$,*

$$\Pr[|\tilde{g}(S) - g(S)| > \epsilon g(S)] \leq 2 \exp\left(-\frac{\epsilon^2 N}{32 \log^2 m}\right).$$

Proof. From Lemma 3.1 part (b), we must have $\gamma^{(1)}$ non-decreasing, hence $\gamma_1^{(1)} \geq 1$. From Lemma E.1, we have

$$g(S) = \alpha_m \mathbb{E} f(X) = \sum_{k=1}^m \beta_k^{(m)} \mathbb{E} f(X_k) \geq \sum_{k=1}^m \frac{k}{m} \beta_k^{(m)} f(S) = \sum_{k=1}^m \frac{\gamma_k^{(m)}}{m} f(S) = \gamma_1^{(1)} f(S) \geq f(S), \quad (16)$$

where the last equality follows from Lemma C.4.

Since $f(X_i) \leq f(S)$ for all i , Hoeffding's bound gives that $\Pr[|\tilde{g}(S) - g(S)| > \epsilon g(S)]$ is at most:

$$2 \exp\left(-\frac{2\epsilon^2 g(S)^2 N}{\alpha_m^2 f(S)^2}\right) \leq 2 \exp\left(-\frac{2\epsilon^2 N}{\alpha_m^2}\right) \leq 2 \exp\left(-\frac{2\epsilon^2 N}{64 \log^2 m}\right),$$

using Lemma E.2. \square

References

- [1] A.A. Ageev and M.I. Sviridenko. Pipage rounding: A new method of constructing algorithms with proven performance guarantee. *J. of Combinatorial Optimization*, 8(3):307–328, September 2004.
- [2] Paola Alimonti. New local search approximation techniques for maximum generalized satisfiability problems. In *CIAC: Proc. of the 2nd Italian Conf. on Algorithms and Complexity*, pages 40–53, 1994.
- [3] Richard A. Brualdi. Comments on bases in dependence structures. *Bull. of the Austral. Math. Soc.*, 1(02):161–167, 1969.

- [4] Niv Buchbinder, Moran Feldman, Joseph (Seffi) Naor, and Roy Schwartz. A tight linear time $(1/2)$ -approximation for unconstrained submodular maximization. In *FOCS*, 2012.
- [5] Gruia Calinescu, Chandra Chekuri, Martin Pál, and Jan Vondrák. Maximizing a submodular set function subject to a matroid constraint (Extended abstract). In *IPCO*, pages 182–196, 2007.
- [6] Gruia Calinescu, Chandra Chekuri, Martin Pál, and Jan Vondrák. Maximizing a monotone submodular function subject to a matroid constraint. *SIAM J. Comput.*, 40(6):1740–1766, 2011.
- [7] Chandra Chekuri, Jan Vondrak, and Rico Zenklusen. Dependent randomized rounding via exchange properties of combinatorial structures. *Proceedings of the 2010 IEEE 51st Annual Symposium on Foundations of Computer Science*, pages 575–584, 2010.
- [8] Jack Edmonds. Matroids and the greedy algorithm. *Math. Programming*, 1(1):127–136, 1971.
- [9] Uriel Feige. A threshold of $\ln n$ for approximating set cover. *J. ACM*, 45:634–652, July 1998.
- [10] Uriel Feige, Vahab S. Mirrokni, and Jan Vondrak. Maximizing non-monotone submodular functions. In *FOCS*, pages 461–471, 2007.
- [11] Moran Feldman, Joseph Naor, and Roy Schwartz. A unified continuous greedy algorithm for submodular maximization. In *FOCS*, pages 570–579, 2011.
- [12] Moran Feldman, Joseph (Seffi) Naor, and Roy Schwartz. Nonmonotone submodular maximization via a structural continuous greedy algorithm. In *ICALP*, pages 342–353, 2011.
- [13] Moran Feldman, Joseph (Seffi) Naor, Roy Schwartz, and Justin Ward. Improved approximations for k -exchange systems. In *ESA*, pages 784–798, 2011.
- [14] Yuval Filmus and Justin Ward. The power of local search: Maximum coverage over a matroid. In *STACS*, pages 601–612, 2012.
- [15] M. L. Fisher, G. L. Nemhauser, and L. A. Wolsey. An analysis of approximations for maximizing submodular set functions—II. In *Polyhedral Combinatorics*, pages 73–87. Springer Berlin Heidelberg, 1978.
- [16] Shayan Oveis Gharan and Jan Vondrák. Submodular maximization by simulated annealing. In *SODA*, pages 1098–1116, 2011.
- [17] Pranava R. Goundan and Andreas S. Schulz. Revisiting the greedy approach to submodular set function maximization. (manuscript), 2007.
- [18] Martin Grötschel, László Lovász, and Alexander Schrijver. *Geometric Algorithms and Combinatorial Optimization*. Springer, 1988.
- [19] Satoru Iwata, Lisa Fleischer, and Satoru Fujishige. A combinatorial strongly polynomial algorithm for minimizing submodular functions. *J. ACM*, 48(4):761–777, 2001.
- [20] Sanjeev Khanna, Rajeev Motwani, Madhu Sudan, and Umesh Vazirani. On syntactic versus computational views of approximability. *SIAM J. Comput.*, 28(1):164–191, 1999.

- [21] Samir Khuller, Anna Moss, and Joseph (Seffi) Naor. The budgeted maximum coverage problem. *Inf. Process. Lett.*, 70:39–45, April 1999.
- [22] Jon Lee, Vahab S. Mirrokni, Viswanath Nagarajan, and Maxim Sviridenko. Non-monotone submodular maximization under matroid and knapsack constraints. In *STOC*, pages 323–332, 2009.
- [23] Jon Lee, Maxim Sviridenko, and Jan Vondrák. Submodular maximization over multiple matroids via generalized exchange properties. *Math. of Oper. Res.*, 35(4):795–806, November 2010.
- [24] G. L. Nemhauser and L. A. Wolsey. Best algorithms for approximating the maximum of a submodular set function. *Math. of Oper. Res.*, 3(3):177–188, 1978.
- [25] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher. An analysis of approximations for maximizing submodular set functions—I. *Math. Programming*, 14(1):265–294, 1978.
- [26] Henri Padé. Sur la représentation approchée d’une fonction par des fractions rationnelles. *Annales scientifiques de l’École Normale Supérieure, Ser. 3*, 9:3–93 (supplement), 1892.
- [27] R. Rado. Note on independence functions. *Proc. of the London Math. Soc.*, s3-7(1):300–320, 1957.
- [28] Jan Vondrák. Optimal approximation for the submodular welfare problem in the value oracle model. In *STOC*, pages 67–74, 2008.
- [29] Jan Vondrák. Submodularity and curvature: the optimal algorithm. In S. Iwata, editor, *RIMS Kokyuroku Bessatsu*, volume B23, Kyoto, 2010.
- [30] Justin Ward. A $(k + 3)/2$ -approximation algorithm for monotone submodular k -set packing and general k -exchange systems. In *STACS*, pages 42–53, 2012.